Analysis of the Power Consumption of a Multimedia Server under Different DVFS Policies

Waltenegus Dargie
Chair for Computer Networks
Faculty of Computer Science
Technical University of Dresden
01062 Dresden, Germany
Email: {waltenegus.dargie}@tu-dresden.de

***Abstract*—Dynamic voltage and frequency scaling (DVFS) has been a useful power management strategy in embedded systems, mobile devices, and wireless sensor networks. Recently, it has also been proposed for servers and data centers in conjunction with service consolidation and optimal resource-pool sizing. In this paper, we experimentally investigate the scope and usefulness of DVFS in a server environment. We set up a multimedia server which will be used in two different scenarios. In the first scenario, the server will host requests to download video files of known and available formats. In the second scenario, videos of unavailable formats can be accepted; in which case the server employs a transcoder to convert between AVI, MPEG and SLV formats before the videos are downloaded. The workload we generate has a uniform arrival rate and an exponentially distributed video size. We use four dynamic scaling policies which are widely used with existing mainstream Linux operating systems. Our observation is that while the gain of DVFS is clear in the first scenario (in which a predominantly IO-bound application is used), its use in the second scenario is rather counterproductive.**

***Index Terms*—Energy consumption of servers, dynamic voltage and frequency scaling, power consumption analysis, dynamic power management, energy consumption**

I. INTRODUCTION

Minimizing the energy consumption of servers and data centers is an active research issue. A universally and steadily increasing energy price and an ever increasing energy budget to accommodate a growing demand for multimedia content storage and sharing necessitates a significant reduction in the energy consumption of the present day ICT infrastructure. Currently a high energy consumption does not necessarily correlate with high performance [8].

Highly adaptive and energy-efficient computing is the only feasible and sustainable solution to this problem. This can be achieved in a number of ways, at various abstraction levels. One of which, and the main focus of this paper, is the use of dynamic voltage and frequency scaling (DVFS) at the operating system level. Proponents of DVFS cite the successes achieved in embedded systems [19], wireless and mobile devices [9], and wireless sensor networks [7] and argue that similar power saving can be achieved with it in servers and data centers.

The central idea of DVFS is that the energy consumption of CMOS-based technology is proportional to the square of the voltage and linearly proportional to the operation frequency [2] [19]. Hence, reducing the voltage of the processor or memory by one fold will reduce the energy consumption of these subsystems by two fold. Likewise, reducing the operation frequency will linearly reduce the energy consumption. There is, of course, a corresponding penalty in performance, but many argue that often Internet-based servers are over provisioned; providing services at 30 to 70% of thier full capacity much of the time [1] [14]. Consequntly, by operating these servers at lower frequencies (as well as voltages) when the workload is less time critical, it is possible to minimize the idle state power consumption, which accounts for more than 50% of the peak power consumption [12], [3] [13].

A substantial body of work exists on DVFS and its application in server environments. A summary of some of the recently proposed approaches can be found in [4]. Most of these approaches claim a significant energy saving. However, most of them are based on analytic or simulation models. Those which are based on real experiment use custom-made workload or existing benchmarks, both of which are criticized for being unrealistic or unrepresentative [16].

A few of these implementations target main stream systems. For instance, Snowdon et al. [21] propose the Koala platform which integrates two DVFS policies. The first one, the maximum-degradation policy chooses the lowest frequency that guarantees a predefined performance threshold. The second policy, the generalized energy delay policy, minimizes the power-delay product ($P^{1-\alpha}T^{1+\alpha}$, where P is the power consumption, T the execution time, and α is a parameter that takes a value between -1 and 1 to set the desired trade-off between energy consumption and performance). The authors integrate the platform into a Linux kernel and their initial test with various benchmarks shows that a significant energy saving can be achieved – up to a gain of 30% for a 3% performance penality.

Likewise, Pallipadi and Starikovskiy [18] at Intel propose the on-demand governor, which can now be integrated into the kernels of many mainstream Linux operating systems. The governor defines utilization thresholds to determine the appropriate CPU frequency. The utilization of the CPU is sampled every x μs, and if the average utilization is below a set threshold (for a given frequency class), then the frequency is reduced by 20%, but if the utilization exceeds the threshold, the CPU frequency is set to maximum. A variant of this governor, the so-called conservative governor, works in a similar fashion with down-scaling; however, a transition from a lower frequency to a higher frequency is a gradual process. The authors experimentally compare the power gains of these two approaches against two settings, namely, when a server runs at maximum frequency (performance state) and when it runs at a low frequency (power-save state). The authors report that, eventhough the actual power gain differs from application to application, the two policies perform better than the performance state with an insignificant penality on performance. We will report a result which contradicts this observation.

In this paper, we experimentally investigate the scope and usefulness of dynamic voltage and frequency scaling in a realistic multimedia server environment. We employ the power-save, on-demand, and conservative policies to scale voltage and frequency. We will use a realistic workload generated on the basis of sound theoretical and probabilistic foundations. We will demonstrate that while it is true that voltage and frequency scaling policies achieve remarkable gain for IO-bound workload, Their effect is generally counterproductive when the CPU has an average workload of 40% and above.

The rest of this paper is organized as follows. In Section II, we outline our methodology to experimentally investigate the scope and usefulness of the power-save, on-demand, and conservative dynamic frequency scaling policies in a multimedia server. In Section III, we analyze the DC power consumption of the server. Finally, in Section IV, we outline some of the open research issues in this area and give concluding remarks.

II. Methodology

A. Cluster Set up

The server cluster we set up for our experiment consists of a load balancer and two multimedia servers connected with each other via a 1 Gbit/s switch. Both servers use Ubuntu Server Edition (v. 10.04)[1]. Moreover Apache[2] is installed on both servers to handle HTTP requests. The clients and the server cluster are isolated from each other by a network emulator (Linktropy 8500) which introduces various constraints (delay, bandwidth, congestion, jitter, and packet loss) into the network. Each server employs an AMD Athlon Dual Core 2 GHz processors, 4 Gbit DDR2 SDRAM memory, 1 Gbit/s network interface card, and 160 GB disk drive. The Apache servers host a large database of videos having different sizes (between 3 MB and 100 MB).

The servers process user requests in two different scenarios. In the first scenario, users request for videos of known and available formats. The servers search these videos and download them without further processing. In the second scenario, the users request for videos of known formats. If the requested formats were not available, then the servers employ a transcoder to carry out format conversion; and then, they download the videos. We use FFmpeg[3] for transcoding the video files.

Each experiment is conducted for one hour. Altogether, we carried out 16 experiments (four times for each scenario). In the subsequent sections, we will analyze the power consumption of one of the servers only, since we have not observed a substantial variation either in resource utilization or in energy consumption between the two of them.

B. Measurement

We employed Yokogawa WT210 digital power analyzers to measure and analyze the energy and power consumptions of the servers. The devices can measure DC as well as AC power consumption at a rate of 10 Hz and a DC current between 15 μA and 26 A with an accuracy of 0.1%.

The load balancer as well as the two servers were built on a D2641 Siemens/Fujitsu Motherboard architecture. The motherboard is supplied with power through two Molex connectors (one 4-pole and the other 25-pole). The 4-pole connector provides a 12 V while the 24-pole provides 3.3 V, 5 V, and 12 V. To understand the DC power consumption of the servers, it necessary to examine how the different operation voltages of the processor and the memory as well as the IO controllers are generated. In fact, the quality of a motherboard is mainly determined by this specific aspect.

The CPU core voltage is generated by a three-phase voltage regulator controlled by an ISL 6312 Pulse Width Modulator (PWM) controller. The PWM controller takes its core voltage from the 5 V line, but the drain of one of the power transistors of the voltage regulator is biased with the 12 V line of the 4-pole connector (we denote this voltage by 12V2). Hence, this line (12 V) is exclusively used by and is responsible for the power consumption of the processor. Likewise, the motherboard provides the memory unit with a single phase voltage regulator controlled by an ISL 6545 PWM controller. This voltage regulator draws much of its current from the 5V line. The motherboard also provides two voltage regulators to the Southbridge and additional voltage regulators to some of the IO controllers. The Southbridge voltage regulators predominantly use the 12 V line of the 24-pole (denoted as 12V1) while all the other IO controllers predominantly draw current from the 3.3 V.

C. Workload Characterization

The workload of a server plays a key role in the final outcome of an experiment. It is, therefore, of profound significance to create a workload that correctly reflects the operation condition of real servers. Le Sueur and Heiser argue that “the SPEC CPU workloads that are commonly used for energy efficiency studies are not representative of the workloads that are run on most real systems. Real systems usually exhibit some level of idleness, allowing CPU sleep states to be used frequently. In contrast, the SPEC CPU workloads are CPU intensive, never allowing the CPU to idle during execution...”

[1]http://www.ubuntu.com

[2]http://www.apache.org/.

[3]http://ffmpeg.org/ (Last visited on February 21, 2012: 15:25 CET.

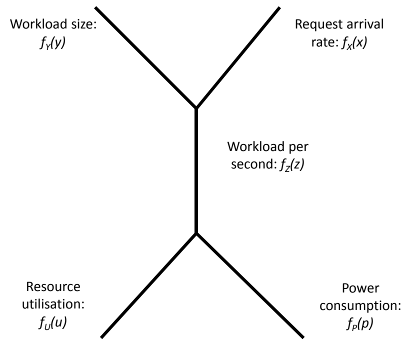

Fig. 1. Request arrival rate, workload size, power consumption, and resource utilization modeled as random variables.

[16]. We agree with this observation and develop a stochastic model to generate a workload to the multimedia servers we set up.

There are several models that characterize the relationship between power consumption, resource utilization, and performance (throughput). Most of these models are deterministic models (for example, see [17], [5], [11]) while a few of them are probabilistic (such as the ones in [13] and [22]). The deterministic models generally assume that the relationship between the three variables can be expressed in terms of properties which are well known and unchanging. The probabilistic models, on the other hand, take the three variables as random variables and use probabilistic tools to analyze their relationship.

In probabilistic models, the workload of a server is a function of the request arrival rate (X) and the workload size each request introduces (Y). The request arrival rate expresses the number of requests received per unit time but ignores the workload each request creates on the server. The second random variable is required to take the workload aspect of each request into account. Therefore, the workload of a server per unit time (Z) is expressed as a multiplication of two random variables, namely, Z = XY. Figure 1 displays the way workload, power consumption, and resource utilization can be modeled as random variables.

The density of Z, $f_Z(z)$ can be expressed using Leibniz's integral rule[4]:

$$f_Z(z) = \int_0^{\infty} \frac{1}{y} f_X(z/y) f_Y(y) dy \quad (1)$$

For example, when both the request arrival rate (X) and the workload of the requests (Y) are uniformly distributed, such that $f_X(x) : U(a,b)$ and $f_Y(y) : U(b,c)$, then the density of Z, $f_Z(z)$, is expressed as [10]:

[4]If $H(z) = \int_{a(z)}^{b(z)} h(x,z)dx$, then, $\frac{d}{dz}H(z) = \frac{db(z)}{dz}h(b(z),z) - \frac{da(z)}{dz}f(a(z),z) + \int_{a(z)}^{b(z)} \frac{\partial h(x,z)}{\partial z}dx$.

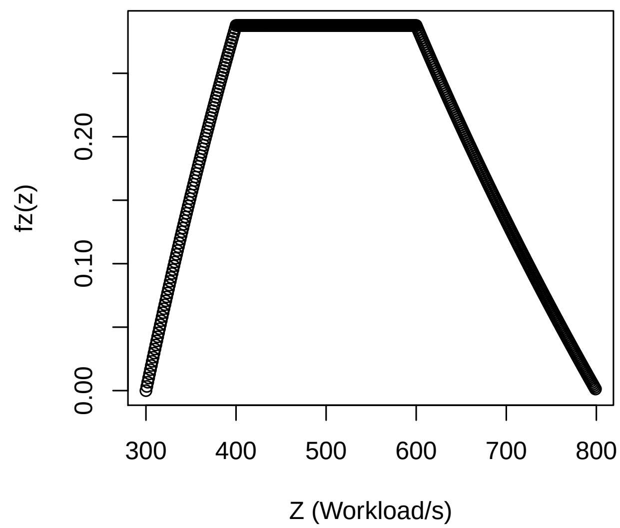

Fig. 2. The pdf of Z (workload per second) when both the arrival rate (number of requests per second) and the workload size (in MB) are uniformly distributed: the first, U(10,20) and the second, U(30,40).

$$f_Z(z) \begin{cases} \int_a^{z/c} \frac{1}{y} f_Y(y) f_X(z/y) dy & ac < z < ad \\ \int_{z/c}^{z/d} \frac{1}{y} f_Y(y) f_X(z/y) dy & ad < z < bc \\ \int_{z/d}^{d} \frac{1}{y} f_Y(y) f_X(z/y) dy & ac < z < bd \end{cases} \quad (2)$$

Figure 2 displays the pdf of Z when $f_X(x) : U(10,20)$ and $f_Y(y) : U(30:40)$.

Similarly, when $f_X(x)$ is uniformly distributed ($f_X(x) : U(a,b)$) and $f_Y(y)$ is exponential ($f_Y(y) : \eta e^{-\eta y}$), $f_Z(z)$ will be:

$$f_Z(z) = \eta \int_0^{\infty} \frac{e^{-\eta u}}{u} du \quad (3)$$

A more realistic model is the one in which the request rates are uniformly distributed while the workload (the size of the videos being downloaded) are exponentially distributed. It has been previously reported that video download requests in IPTV users follows a power distribution [15]. We used this model to investigate the relationship between workload, resource utilization, and power consumption. The user request arrival rate varies between 0 and 100 requests per second, while the video size varies between 3 and 100 MB, with a mean video size (η) of 3 MB.

D. Server Runtime Characteristic

As mentioned earlier, we carry out our experiment in two different scenarios: when Apache is running with and without a transcoder. The transcoder converts requested videos between AVI, MPG4, and SLV formats before the videos are ready for download.

We integrated the **cpufrequtils** utilities[5] into the Ubuntu kernel infrastructure for supporting DVFS. The utilities pro-

[5]https://wiki.archlinux.org/index.php/Official_Repositories: Last accessed on November 14, 2011: 22:38 CET.

vide us with three different types of policies: power-save, on-Demand, and conservative. The power-consumption and performance of the servers under these policies will be compared to each other and to the condition in which no scaling policy is used, i.e., when the servers are running at maximum frequency. The latter state is called *performance state*. The core voltage (VDD) of the AMD Athlon Dual Core processors can be varied between 0.8 V and 1.55 V in step of 0.25 V. Hence, the permissible core voltages are: 0.8 V, 1.05 V, 1.3 V and 1.55 V. The corresponding operation frequencies for these voltages are 1000 MHz, 1330 MHz, 1670 MHz, and 2000 MHz.

The power-save policy operates the processors at the lowest frequency while the on-demand and conservative policies adapt the clock frequency to the change in the workload of the servers. The operating system enables to sample the CPU utilization between 10700 and 4294967295 μs. When the sampling period is too short, the sampling overhead will become too high; when it is too long, then the estimation will be inaccurate. We choose 100 ms to achieve a balance between estimation latency and estimation overhead.

The Apache server is IO-intensive while the transcoder is CPU-intensive. Therefore, we expect that dynamic voltage and frequency scaling will be more effective when the servers run Apache alone. Furthermore, due to the complementary nature of the two services, the power efficiency will be high when they are consolidated on a single server rather than when they run on separate servers. However, each service has a tendency to quickly saturate different resources (Apache saturating the network bandwidth whereas FFmpeg saturating the CPU), in which case, it is unavoidable to run them on separate servers when the workload reaches a certain level.

III. Power Consumption Analysis

A. Overall Power Consumption

Figure 3 displays the cumulative distribution function (CDF) of the overall power consumption of the multimedia server when Apache was the only service running on it. The corresponding CPU utilization is displayed in Figure 5 (left). Table I displays the throughput (in GB) of the server for one hour. As expected, the server's power consumption and CPU utilization reached the highest level when it operated at the maximum frequency, but the highest throughput was also obtained with the highest frequency. In the performance state, the server consumed on average 58.5 W whereas it consumed 51.3 W with the conservative, 53 W with the on-demand, and 51.9 W with the power save policies. The variance of the power consumption of the different power management policies was 10.89 (standard deviation = 3.3). The variance of the throughput was 11.35 (standard deviation = 3.4). From the variances, it can be stated that an increment (decrement) in power consumption resulted in a corresponding increment (decrement) in performance.

The power consumption of the server and the CPU utilization increased significantly when the server run both services. As expected, the throughput reduced significantly, due to the transcoding process. Interestingly, the server consumed a higher amount of power under the on-demand and conservative policies than when it operated at maximum frequency. The average power consumption of the server when it operated under the performance state was 73.3 W while it was 78.1 W under the conservative policy and 81.8 W under the on-demand policy. For the power save policy, the average power consumption was 62.4 W. The variance of power consumption under the different frequency scaling policies when the two services run together was 71 (standard deviation = 8.43). The corresponding variance of the throughput was 15.96 (standard deviation = 3.99). The difference between the variance of the power consumption and the throughput is notably high in this case. Moreover, the variance of the power consumption is higher than the variance of the throughput, clearly indicating that varying the power consumption does not lead to a corresponding variation in throughput.

What is interesting still is that the throughputs of the on-demand and the conservative policies are lower than the throughput of the performance policy, even though both policies resulted in higher power consumptions than the performance policy. This clearly indicates that in the second scenario (when the two services run at the same time) the cost of dynamic voltage and frequency scaling outweighs the gain that can be achieved by it.

In the first scenario (when Apache was running alone), the processor of the server were much of the time idle. For

Policy	Without Transcoder	With Transcoder
Performance	49.87089	1.362283
Conservative	55.24799	0.6737399
On-demand	51.80392	0.9463543
Power save	50.17798	0.5701639

TABLE I
COMPARISON OF THE THROUGHPUT (IN GB) OF THE MULTIMEDIA SERVER WITH AND WITHOUT TRANSCODING.

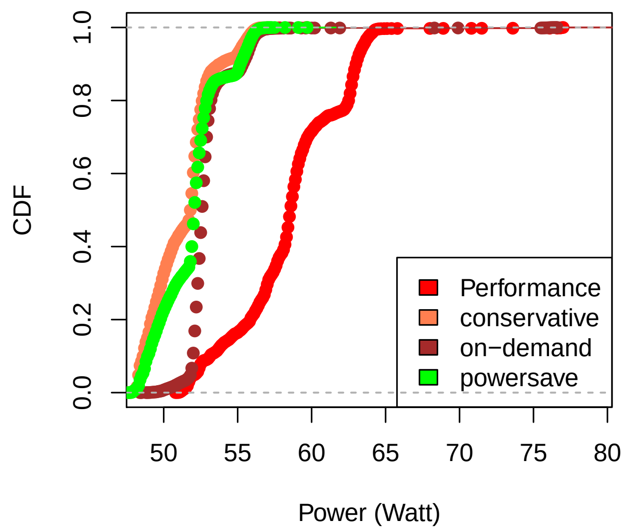

Fig. 3. The overall power consumption of one of the multimedia servers when it runs Apache only.

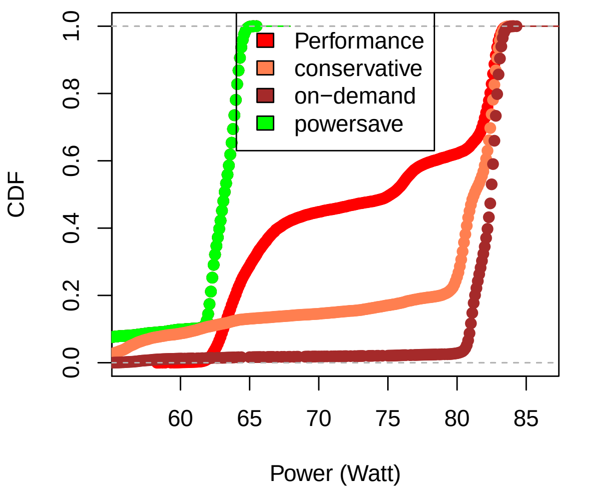

Fig. 4. The overall power consumption of one of the multimedia servers when it runs both Apache and the FFmpeg transcoder.

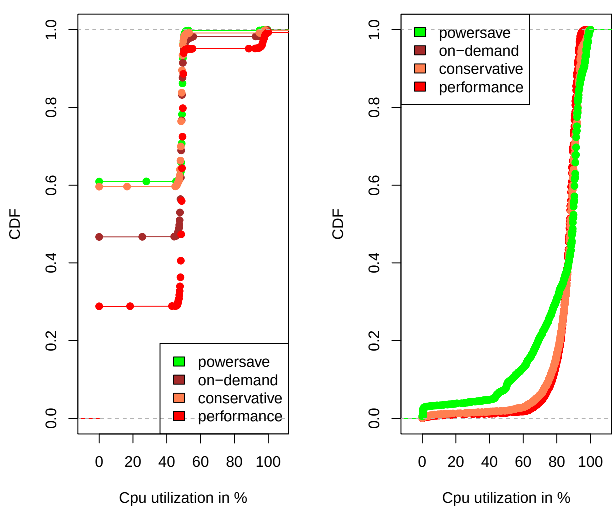

Fig. 5. CPU utilization with and without the transcoder

example, under the power save policy, $P(U \leq 40\%) = 0.61$ and under performance, $P(U \leq 40\%) = 0.3$. As a result, all the frequency scaling policies were comparatively effective. Unlike the first scenario, in the second scenario, the CPU utilisation in all the frequency scaling policies was high. Exceping the power save policiy, $P(U \leq 40\%) \approx 0$. Under all circumstances, $P(U \leq 80\%) \geq 0.6$. from this, it is reasonable to conclude that frequency scaling policies do not produce any appreciable gain when the CPU utilization of the server is above 80% much of the time. In fact, the penality of switching the CPU frequency from one level to another is high, resulting in a high overall power consumption. Consequently, any gain in the power consumption is achieved with a considerable performance penality (as is the case with the power save policy).

B. DC Power Analysis

To better understand the power consumption characteristics of the server under the different frequency scaling policies, we take a closed look at the DC power consumption. It has a static and a dynamic aspect. The DC power consumed through the 3.3 V and the 12V1 (Figure 6) do not change much under the various settings (configurations). Likewise, the power consumption of the disk drive can be considered as a constant cost[6].

This is consistent with our analysis of the power distribution in the D2461 motherboard (Section II-B). The 3.3 V line supplies power to the peripherals (including the NIC and the graphic card). The power consumption of the NIC is around 2 W and remains invariable throughout the experiment. The remaining power consumption is on account of the graphic card, which is also appreciably small. The 12V1 is used essentially as a control signal by the voltage regulators of the Southbridge and the memory termination logic. Similarly, the CPU fan is supplied with power through the 12V1 line and it is both small and invariable.

The power drawn through the 5.5 V and 12V2 shows a visible change following the change in the workload of the server. For example, Figure 7 displays the power consumption of the server when it operates at maximum frequency in both scenarios. In the first case, the average AC power consumption of the server was about 52 W. The average DC power consumption of the server observed from all the DC supply lines was 36 W. Hence, 16 W (30%) was lost due to the inefficiency of the power supply unit, which is consistent with the ATX specification[7]. At this load, the ATX specification requires that the power supply unit should have an efficiency of 65%. The power efficiency of the power supply unit improves to 70% when the server operates at maximum frequency and when the two services were running at the same time, in which case, the average overall AC power consumption of the server was 82 W and the average DC power consumption was 57.22 W.

As can be seen from Figure 7, a significant portion of the DC power consumption of the server was on account of the processor. Moreover, this power consumption is dynamic, since it varied according to the workload of the server. Much of the power consumed via the 5 V line is due to the memory subsystem and the memory termination logic, since the power transistors of the voltage regulator of the memory subsystem draw current through this line. This power consumption exhibited two aspects when the server was running Apache: a static and a dynamic aspect. The 5 W power consumption was the minimum power required to power the memory modules. The additional power consumption (approx. 3 W), was due to the transfer of data between memory and the CPU

[6]This is true as long as the request rate is below 100/s and the average file size is 3 MB. For larger request rates and larger video data, the power consumption of the disk drive fluctuates between 7 and 14 W.

[7]ATX Specification, version 2.2 (2003 – 2005), Intel Corporation.

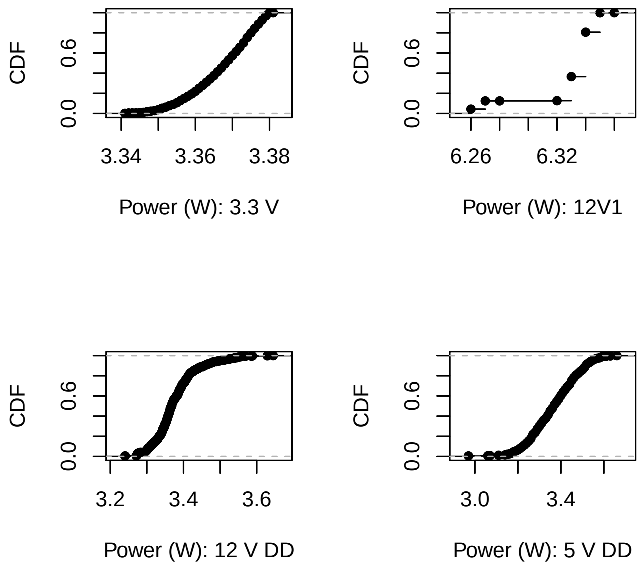

Fig. 6. The constant DC power costs of the multimedia server.

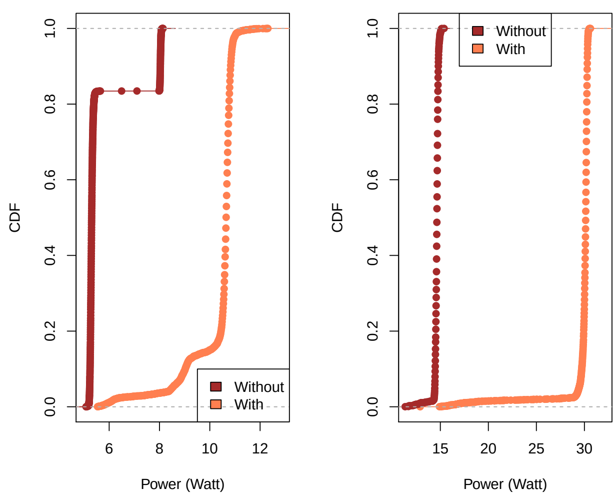

Fig. 7. The power drawn by the server from the 5 V (left) and 12V2 (right) power lines when the server operated at maximum frequency.

(VTT) and it is a dynamic cost. The dynamic aspect becomes dominant when the two services run at the same time, since the transconder is a memory-intensive service. The difference in power consumption between the four policies, as far as the power consumed through the 5 V line is concerned, was not big – a maximum difference of 2 W. This is shown in Figure 8. This observation may suggest that DVFS may not work well for the memory subsystem.

IV. Discussion

We experimentally analyzed the power consumption of a multimedia server under different dynamic voltage and fre-

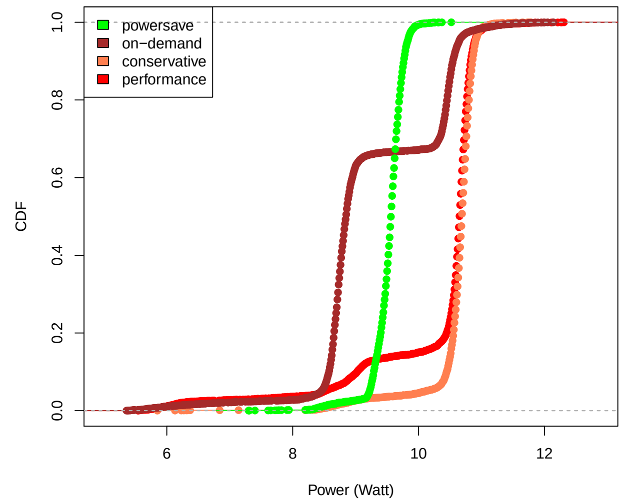

Fig. 8. The dynamic power drawn through the 5 V line (much of this is on account of the memory subsystem).

quency scaling policies. The policies are exemplary in that they represent four different approaches. The first two set the frequency of a processor at a maximum or minimum level to reduce the cost of switching between different frequency levels. The other two approaches estimate the future operation frequency by observing the utilization (which corresponds to the workload) of the processor for the past x μs. One of them, the on-demand policy, enables the processor to make a directly transition from any of the lower frequency levels to the highest frequency level while transition in operation frequency is gradual in the conservative policy.

Each of these policies have their merits and demerits. Setting the frequency of the CPU at a fixed value makes sense when the workload is static or predominantly static in nature. Otherwise, power saving will only be achieved at a cost of performance. The justification of the conservative policy is stability, since a drastic change in the frequency requires a corresponding adjustment in the core voltage. The voltage regulator in turn requires certain amount of time to generate the desired voltage. For example, the ISL 6312 voltage regulator has 2.063 $mV/\mu s$ soft-start ramp rate when it supplies the AMD processor with power. Therefore, as the transition gap increases, the time required by the regulator to generate the desired voltage increases. The CPU may become unstable during this period. The justification for the on-demand policy, on the other hand, is that the conservative policy may not be appropriate for bursty workload – by the time the CPU frequency reached maximum, the workload profile has already changed.

A typical Internet workload has both dynamic and static aspects and, hence, no single policy is equally effective. Moreover, appreciable gain can only be achieved if the CPU operates at a given frequency for a certain period of time. This duration has been analytically computed [6], [7], but there are two problems with it. Firstly, the analytic models assume that the power consumption of a workload can be known in a deterministic way. Secondly, even if it can be assumed that a deterministic relationship between the workload and the power consumption exists, predicting the workload is not a trivial assignment. Sinha and Chandrakasan [20] experiment with different types of filters (moving average, exponential, weighted average least mean square) to predict a workload and find out that the least mean square filter performs better than all the others. However, this filter requires more computation and resources than all the others. Consequently, for a highly fluctuating workload, the estimation cost is high while achieving estimation becomes difficult. For all these reasons, the ondemand and conservative policies performed poorly when the Apache server and the transcoder run on the multimedia server at the same time. They neither reduce the power consumption not achieved appreciable throughput.

We would like to compare the policies presented in this paper with other policies, such as Koala [21], but obtaining the implementation of the policies was not possible. In the future, we aim to include additional policies in out investigation.

ACKNOWLEDGMENT

This work has been partially funded by the German Research Foundation (DFG) under project agreement SFB 912/1 2011. Ms. Alaa Brihi was responsible for undertaking a part of the experiment presented in this paper.

REFERENCES

[1] F. Ahmad and T. N. Vijaykumar. Joint optimization of idle and cooling power in data centers while maintaining response time. In *Proceedings of the fifteenth edition of ASPLOS on Architectural support for programming languages and operating systems*, ASPLOS '10, pages 243–256, New York, NY, USA, 2010. ACM.

[2] H. Aydin, V. Devadas, and D. Zhu. System-level energy management for periodic real-time tasks. In *Proceedings of the 27th IEEE International Real-Time Systems Symposium*, pages 313–322, Washington, DC, USA, 2006. IEEE Computer Society.

[3] L. A. Barroso and U. Hölzle. The case for energy-proportional computing. *Computer*, 40:33–37, December 2007.

[4] L. Bertini, J. C. B. Leite, and D. Mossé. Power optimization for dynamic configuration in heterogeneous web server clusters. *J. Syst. Softw.*, 83:585–598, April 2010.

[5] Y. Chen, A. Das, W. Qin, A. Sivasubramaniam, Q. Wang, and N. Gautam. Managing server energy and operational costs in hosting centers. *SIGMETRICS Perform. Eval. Rev.*, 33:303–314, June 2005.

[6] C. Chiasserini and R. Rao. Improving energy saving in wireless systems by using dynamic power management. *IEEE Transactions on wireless communications*, 2(5):1090–1100, 2003.

[7] W. Dargie. Dynamic power management in wireless sensor network: State-of-the-art. *IEEE Sensor Journal*, 12(5):1518–1528, 2012.

[8] W. Dargie, A. Strunk, and A. Schill. Energy-aware service execution. In *The 36th Annual IEEE Conference on Local Computer Networks*, 2011.

[9] G. Dhiman and T. S. Rosing. Dynamic voltage frequency scaling for multi-tasking systems using online learning. In *Proceedings of the 2007 international symposium on Low power electronics and design*, ISLPED '07, pages 207–212, New York, NY, USA, 2007. ACM.

[10] J. H. Drew, A. G. Glen, and L. M. Leemis. Computing the cumulative distribution function of the kolmogorov-smirnov statistic. *Comput. Stat. Data Anal.*, 34:1–15, August 2000.

[11] D. Economou, S. Rivoire, and C. Kozyrakis. Full-system power analysis and modeling for server environments. In *The 2nd Workshop on Modeling, Benchmarking, and Simulation (MoBS)*, pages 70–77, 2006.

[12] X. Fan, W.-D. Weber, and L. A. Barroso. Power provisioning for a warehouse-sized computer. In *ISCA '07: Proceedings of the 34th annual international symposium on Computer architecture*, pages 13–23, New York, NY, USA, 2007. ACM.

[13] A. Gandhi, V. Gupta, M. Harchol-Balter, and M. A. Kozuch. Optimality analysis of energy-performance trade-off for server farm management. *Perform. Eval.*, 67:1155–1171, November 2010.

[14] J. Hamilton. Internet-scale service infrastructure efficiency. *SIGARCH Comput. Archit. News*, 37:232–232, June 2009.

[15] X. Hei, C. Liang, J. Liang, Y. Liu, and K. Ross. A measurement study of a large-scale p2p iptv system. *IEEE Transactions on Multimedia*, 9(7):1672–1687, 2007.

[16] E. Le Sueur and G. Heiser. Slow down or sleep, that is the question. In *Proceedings of the 2011 USENIX conference on USENIX annual technical conference*, USENIXATC'11, pages 16–16, Berkeley, CA, USA, 2011. USENIX Association.

[17] A. Lewis, S. Ghosh, and N.-F. Tzeng. Run-time energy consumption estimation based on workload in server systems. In *Proceedings of the 2008 conference on Power aware computing and systems*, HotPower'08, pages 4–4, Berkeley, CA, USA, 2008. USENIX Association.

[18] V. Pallipadi and A. Starikovisky. The ondemand governer. In *Proceedings of the Linux Symposium (volume two)*, 2006.

[19] P. Pillai and K. G. Shin. Real-time dynamic voltage scaling for low-power embedded operating systems. *SIGOPS Oper. Syst. Rev.*, 35:89–102, October 2001.

[20] A. Sinha and A. Chandrakasan. Dynamic power management in wireless sensor networks. *IEEE Des. Test*, 18(2):62–74, 2001.

[21] D. C. Snowdon, E. Le Sueur, S. M. Petters, and G. Heiser. Koala: a platform for os-level power management. In *Proceedings of the 4th ACM European conference on Computer systems*, EuroSys '09, pages 289–302, New York, NY, USA, 2009. ACM.

[22] Q. Zhu, J. Zhu, and G. Agrawal. Power-aware consolidation of scientific workflows in virtualized environments. In *Proceedings of the 2010 ACM/IEEE International Conference for High Performance Computing, Networking, Storage and Analysis*, SC '10, pages 1–12, Washington, DC, USA, 2010. IEEE Computer Society.