A Comparison of High-Level Full-System Power Models

Suzanne Rivoire
Sonoma State University

Parthasarathy Ranganathan
Hewlett-Packard Labs

Christos Kozyrakis
Stanford University

Abstract

Dynamic power management in enterprise environments requires an understanding of the relationship between resource utilization and system-level power consumption. Power models based on resource utilization have been proposed in the context of enabling specific energy-efficiency optimizations on specific machines, but the accuracy and portability of different approaches to modeling have not been systematically compared. In this work, we use a common infrastructure to fit a family of high-level full-system power models, and we compare these models over a wide variation of workloads and machines, from a laptop to a server. This analysis shows that a model based on OS utilization metrics and CPU performance counters is generally most accurate across the machines and workloads tested. It is particularly useful for machines whose dynamic power consumption is not dominated by the CPU, as well as machines with aggressively power-managed CPUs, two classes of systems that are increasingly prevalent.

1 Introduction

In order to maximize energy efficiency, whether in a single system or over an ensemble of systems, users and data center operators need to understand the relationship between resource usage and system-level power consumption. This understanding enables such optimizations as consolidating workloads on as few machines as possible and turning others off [6, 16], since current hardware is highly inefficient at low utilization [1]. It also includes dynamically adjusting power budgets within an enclosure or rack to enable a data center's power provisioning infrastructure to be designed less conservatively [5, 18]. While several types of full-system power models have been proposed, often in the context of enabling a particular optimization, they have not been systematically compared over a variety of software workloads and hardware platforms.

The desirable properties of a full-system model suitable for this type of online use are as follows:

Accuracy: The model must be accurate enough to allow the desired energy savings. In this work, we define accuracy as less than 10% error in the average case, although higher accuracies are certainly better and may be necessary for some optimizations.

Speed: The model must generate predictions quickly enough to be used by energy-saving policies; this work assumes that one sample per second is sufficiently fast.

Generality and **portability:** The modeling framework should be suitable for as many systems as possible, encompassing mobile, desktop and server systems; different processor families and memory and storage technologies; and different power footprints and dynamic ranges. It should also allow for different allocations of power consumption among components, rather than assuming that a particular component will dominate the system. Furthermore, generating models for a new system should not require exhaustive design space exploration or tuning.

Inexpensive: Generating and using the model should not require expensive or intrusive infrastructure.

Simple: All other things being equal, simple models should be preferred to complex ones, both in terms of the number of inputs and the complexity of the relationship between inputs and outputs.

In this work, we use a common infrastructure to generate several high-level full-system power models for a wide range of workloads and machines, and we compare the accuracy of these models in each instance. This analysis illuminates the trade-offs between simplicity and accuracy, as well as the limitations of each type of model. We begin with a discussion of previously proposed approaches to power modeling in Section 2, before defining the models compared in this work and describing the infrastructure used to generate them in Section 3. Section 4 then describes the software and hardware platforms used for their evaluation. Section 5 presents the results of the evaluation, which are discussed qualitatively in Section 6. Section 7 presents the overall conclusions and directions for future work.

2 Related Work

Approaches to real-time power modeling fall into two categories: detailed analytical power models, and high-level black-box models. Detailed analytical processor power models based on CPU performance counters have been proposed by Joseph and Martonosi [9] and by Isci and Martonosi [8]. These models are real-time and highly accurate, but they model only the processor component, and they rely on detailed microarchitectural knowledge of a particular processor, limiting their generality and portability. Kadayif *et al.* used performance counters to model the energy consumption of the Sun UltraSPARC memory hierarchy, with the goal of synthesizing this information into virtual "energy counters" [10]. These models all provide a middle ground between highly detailed simulation-based models and the simple real-time models evaluated in this paper.

The other type of real-time power model is the high-level, black-box model, which sacrifices some accuracy in order to avoid relying on detailed knowledge of the hardware's implementation. Such models have been developed for processors, single systems, and groups of systems in enterprise environments. At the processor level, Bellosa's power models were some of the first to correlate power consumption with performance counters, finding linear relationships between processor power consumption and several performance counters [2]. Contreras and Martonosi created a highly accurate black-box linear model of the power consumption of an Intel XScale processor and its memory system [3]. These models are simple, fast, and low-overhead, but they do not model the full-system power consumption, and it is unclear how general and portable they are.

At the full-system level, Li and John used performance counters to generate models for individual OS routines, concluding that models using performance counters without higher-level software knowledge were not consistently accurate [11]. Cignetti *et al.* developed a full-system energy model for a Palm IIIE PDA as a linear function of the power states of its eight major components. In the server environment, Ranganathan *et al.* implemented dynamic power budgeting optimizations [18] by creating lookup tables relating power and performance to system resource utilization, and they developed a methodology for porting a model generated on one machine to another in the same family [17]. Fan *et al.* implemented similar optimizations using models based on CPU utilization alone: one linear, and

the other using an empirical power term [5]. To facilitate server consolidation, Heath *et al.* developed linear models that used OS-reported CPU, memory, and disk utilization for two different types of servers [6]. Finally, Economou *et al.* used OS-reported component utilizations and CPU performance counters to model the power consumptions of a blade and an Itanium server [4]. This study evaluates the latter approaches for a wider range of systems and workloads.

3 Model Development

To develop the power models for each system, we use Mantis [4], a non-intrusive infrastructure for providing fast and accurate full-system power predictions. Mantis requires a one-time model fitting, in which the system is plugged into an AC power meter, and power readings and software metrics are collected once per second while the system runs a special calibration workload. The software metrics are then fitted to the power readings, and can be used thenceforth to predict power without the presence of a meter. The special calibration workload includes programs that selectively stress each of the major consumers of dynamic power (CPU, memory, and disk) at varying levels of utilization [13, 19]. The network subsystem is not yet a noticeable consumer of dynamic power [4, 14], and thus was not stressed by the calibration suite. This approach to calibration, stressing each component in isolation, implicitly assumes that the components are linearly independent; this assumption restricts the complexity of the models that can be generated. Furthermore, the calibration suite is biased toward the idle case; if the CPU, memory, and disk tests take equal amounts of time, each component is idle for two-thirds of the calibration suite, and we do not correct for this. The benchmarks that we use to evaluate the models in Section 5, however, operate at high utilization and will thus illustrate the worst-case errors.

3.1 Model inputs

We compare five different types of models in this work, which vary in the inputs used and the complexity of the model equations. The first model simply predicts a constant power C_0, regardless of resource utilization; this C_0 is the mean power during the calibration suite. This model is valuable for two reasons: first, it provides a baseline or null model for the utilization-based models. Second, it is similar to the method of estimating a system's power based on its manufacturer specifications, although it will probably yield better predictions, since it is the average power during use rather than a conservative estimate. Using only full-system power measurements, as we do here, it is impossible to delineate each component's contribution to the idle power C_0; we can only model the change in power correlated to each component's utilization.

The second and third models, proposed by Fan *et al.* [5] use only CPU utilization as input to the model. The first of these is linear in CPU utilization, yielding an equation of the form $P_{pred} = C_0 + C_1 \times u_{cpu}$. The second adds an empirical term that those authors found to improve accuracy, yielding an equation of the form $P_{pred} = C_0 + C_1 \times u_{cpu} + C_2 \times u_{cpu}^r$, where C_2 and r are additional fitted parameters.

The fourth model, similar to that proposed by Heath *et al.*, is a linear model that uses disk utilization in addition to CPU utilization [6]. To model the dynamic power consumption of the disk, we also evaluated models that used the number of I/O requests and the number of disk transfers as parameters in order to have some idea of the balance of random vs. sequential I/O. However, these models were no more accurate than those simply using disk utilization, which we present here. The final model, which we proposed [4], adds CPU performance counters to the OS-reported CPU and disk utilization. In the interest of simplicity and low overhead, we use only as many counters as can be collected at one time, without time-multiplexing; for most of our systems, the maximum number of counters is 4. In general, the counters we examined correspond to the memory bandwidth, the amount of instruction-level parallelism, the activity of the cache hierarchy, and the utilization of the floating-point unit. The exact counters available are highly implementation-dependent, as are the weights assigned to them in the models.

4 Evaluation Methodology

To evaluate the generality of the models, we used a wide variety of machines and benchmarks. The machines studied were:

- An 8-core (2 quad-core) Xeon server with 32 GB FBDIMM memory
- A server with 4 Itanium 2 CPUs and 1 GB memory
- A mobile fileserver with a high-end mobile CPU and 13 SATA laptop disks [20], at its highest and lowest CPU clock frequencies
- The same machine with just one disk, at its highest and lowest CPU clock frequencies
- A 2005-era AMD Turion laptop with 384 MB of memory

These machines span three different processor families (Core 2 Duo/Xeon, Itanium, and Turion) from two different manufacturers. Their memories include mobile memory, desktop/server DDR2 memory, and FBDIMM technology, and their disks include both mobile and enterprise disks. At the full-system level, the machines vary in the balance of power among their components and in the amount of variation in their dynamic power consumption, from the CPU-dominated Itanium server to the disk-intensive mobile fileserver.

The benchmarks used were:

- The SPEC CPU integer and floating-point suites
- The SPEC JBB benchmark
- The "stream" memory-stress synthetic benchmark [12]
- I/O-intensive benchmarks, which varied by system. For most machines, the ClamAV antivirus scanner was used, with multiple copies instantiated if necessary to exercise multiple disks. Two machines required different programs: on the mobile fileserver, ClamAV saturated the CPU before the disk subsystem, so the Nsort [15] sorting program was used instead. For the Itanium, the SPECweb benchmark was used to provide a combination of disk and network I/O.

We evaluate the models by both generating model predictions and measuring the full-system (wall) power with a power meter, and then comparing the predictions with the measured power. These measurements and comparisons were also done once per second, which is the limit of many interfaces to system utilization metrics (*e.g.* perfmon, sar, etc.). This granularity is sufficient for the cross-system optimizations described in Section 1.

5 Results

Figure 1 shows the mean absolute percentage error for each model across all of the machine configurations tested. The 90^{th} percentile absolute error, which is omitted here for brevity, is not qualitatively different and is documented in [19]. The first cluster of columns shows the error for the calibration suite that was used to fit the models; the remaining clusters show the error for each of the benchmarks described in the previous section. Within each cluster, the leftmost bar denotes the error from the constant (utilization-independent) power model, followed by the linear and empirical CPU-utilization-based models, the linear CPU- and disk-utilization-based model, and finally the

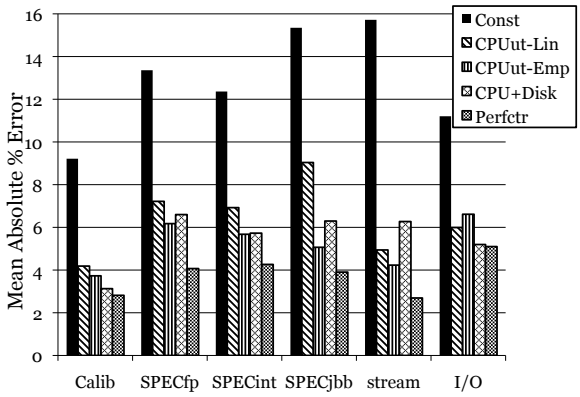

Figure 1: Overall mean absolute error for the generated models over all benchmarks and machine configurations.

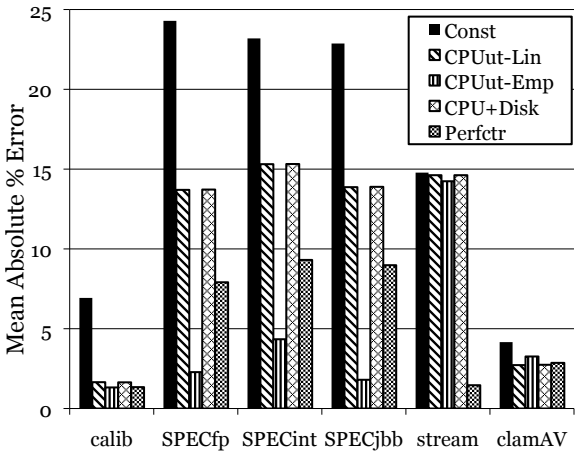

Figure 2: Mean absolute percentage error for the generated models on the Xeon server.

linear model using CPU performance counters in addition to the OS-reported metrics.

These aggregate results show that all of the utilization-based models are superior to the constant model, showing that resource utilization metrics correlate to power consumption. The most detailed model, which is the performance-counter-based model, is the best model overall for every benchmark, with the lowest mean and 90^{th} percentile absolute errors. However, all of the models predict power within 10% mean accuracy for each benchmark, averaged over all configurations. These results suggest that even the simple, linear CPU-utilization-based model would meet the accuracy goal outlined in Section 1. The rest of this section examines the results for some of the individual machines, showing where more detailed models are most necessary; again, the full results for all machines, including the full model equations, are documented in [19].

5.1 Xeon server

Figure 2 shows the mean absolute percentage error for the different models on the Xeon server. All of the utilization-based models fit the calibration suite, which is biased toward the idle case, with very little error. However, the errors for the SPEC benchmarks and stream are much higher. In the case of the SPEC benchmarks, which are CPU-intensive, the performance-counter-based model is the best of the linear models, but the linear models are all outperformed by the empirical CPU-utilization-based model. Figure 3 shows the measured power

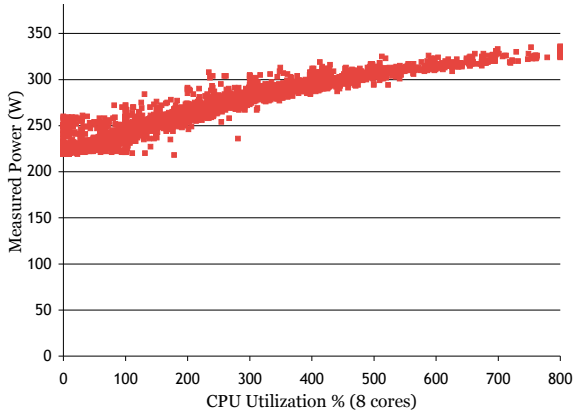

Figure 3: Measured power versus CPU utilization for the 8-core Xeon server during the calibration suite. Note that memory and disk utilization were not constant over this dataset and that the calibration suite does not fully stress all components simultaneously.

during the calibration suite versus CPU utilization, a curve that is close to linear for utilizations up to 500% (5 cores) and then leveling off. This behavior is captured by the empirical equation and may be a result of the shared resources on the two quad-core Xeon chips used in this server; we found that the maximum number of references to the shared L2 cache increases very slowly at high utilization, since this resource can be highly utilized by just two cores. This empirical model was originally proposed in a multicore server environment [5], and its ability to model bottlenecks on shared resources is well suited to these processors.

The stream benchmark's power consumption is best modeled by the performance-counter-based models. The Xeon server has 32 GB of FBDIMM memory, which has a dynamic power variation of up to 100 W. Since the stream benchmark is memory-intensive without commensurate CPU activity, and since only the performance-counter-based model uses memory activity as a parameter, it is no surprise that the performance-counter-based models are most accurate. Finally, the models are all quite accurate for the disk-intensive ClamAV benchmark, since the disk contributes a very small fraction of the system's dynamic power consumption.

5.2 Mobile fileserver

Figures 4 and 5 show the mean absolute percentage error of the mobile fileserver at its highest and lowest clock frequencies (separate models were generated for each frequency). For the SPEC benchmarks, the disk- and performance-counter-based models perform much better than the models based on CPU utilization alone. The usefulness of disk information is counter-intuitive, since neither of these benchmarks has significant disk utilization. The reason that the disk information is helpful is that both CPU and disks contribute highly (and about equally) to the dynamic power of this system, and they are not highly correlated to each other. Forcing the CPU utilization to explain the entire dynamic power variation results in over-predicting the power consumption when the disk utilization is low, particularly for the less flexible linear model. Performance counter information further improves the power predictions; the reason is that this CPU uses aggressive clock gating to shut down unused units, even at high utilization [7], so power depends on how the CPU is utilized—-a more complex question than whether it is active at all, which is what CPU utilization reports. This effect is particularly pronounced with SPECjbb, which runs at full

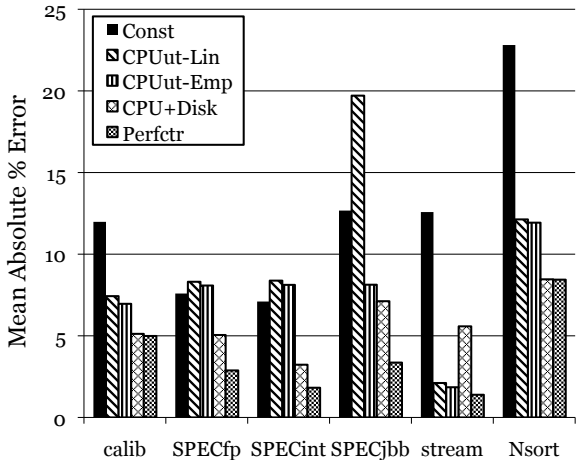

Figure 4: Mean absolute percentage error of the generated models on the mobile fileserver at its highest frequency (2.3 GHz).

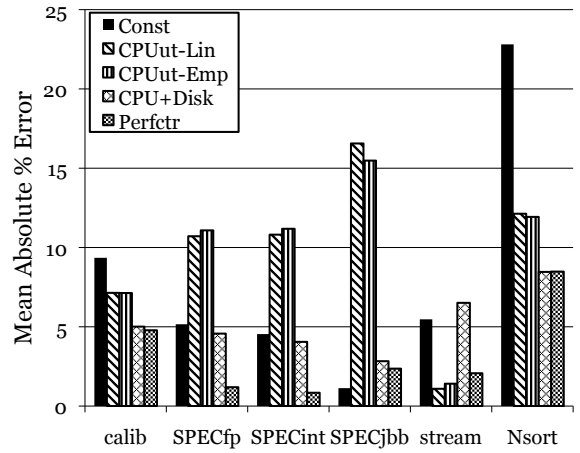

Figure 5: Mean absolute percentage error of the generated models on the mobile fileserver at its lowest frequency (1 GHz).

utilization but with much lower instruction-level parallelism than SPECint and SPECfp, even though we ran SPECjbb on two cores and the CPU suite on just one.

The accuracies at the lowest frequency are much lower than at the highest frequency. The reason is that the CPU becomes a smaller consumer of dynamic power at this frequency; its dynamic power is dwarfed by the disk's and is slightly lower than the memory's. The CPU-utilization-based models, which are forced to attribute the entire dynamic power variation of the system to the component with the lowest variation, yield predictions worse than the constant model for the CPU-intensive benchmarks; paradoxically, they predict the power well for stream, since their overpredictions of CPU power end up being fairly close to the dynamic memory power. Overall, this configuration shows the difficulty of developing power models for systems where the CPU does not dominate the dynamic power. One reason for this difficulty is the lack of detailed memory and disk metrics analogous to CPU performance counters, which would help model these components' dynamic power consumption.

6 Discussion

The results presented in the previous section show that simple, high-level models can be used to accurately predict power for a wide range of systems and workloads. They also highlight some potential pitfalls in developing power models:

- **CPU utilization as a proxy for power consumption.** CPU utilization is often considered a first-order proxy for dynamic power consumption, the logic being that the CPU is the dominant consumer of dynamic power and that its power is determined largely by its power state (active or sleeping). For systems that are not CPU-dominated (*e.g.* fileservers, some laptops) or workloads that are not CPU-intensive (*e.g.* streaming, sorting), these assumptions break down. They also break down in two other situations, even for systems and workloads that are CPU-dominated. The first is in processors with shared resources, such as multicore processors, where power consumption is not a linear function of utilization. The second is in aggressively power-managed processors, where the active-state power consumption may vary widely depending on what the CPU is actually doing. Since current hardware trends are toward less CPU-dominated systems [1], multicore processors, and aggressive power management, OS-reported CPU utilization will be a less and less useful proxy for power consumption.
- **Assuming that more information yields a better model.** If a large component of the system's dynamic power is not directly accounted for by the power models, less detailed models may actually yield better results, as seen with the stream benchmark on the mobile fileserver. In general, models' relative accuracy will be unpredictable if a large component of dynamic power is not modeled.
- **Blindly applying performance counters across systems.** While the same basic group of performance counters (unhalted clock cycles, instructions retired/ILP, last-level cache references, memory bandwidth, and floating-point instructions executed) can be used to model power across a wide variety of systems, the nuances of how these counters are defined from platform to platform do matter. To yield an accurate model, a performance event must be represented over its entire range in the calibration suite. When generating models for the laptop, which has an AMD Turion processor, the distinction between AMD's "memory requests" counter and Intel's "memory bus transactions" for the purpose of power modeling became significant. Counting the number of memory requests does not account for the activity of the prefetcher, although all memory traffic contributes to power consumption. Because the memory accesses in the calibration suite are highly regular, the "memory requests" counter could not be used to generate accurate models.
- **Lack of insight into memory and disk power**. Systems that were CPU-dominated resulted in more accurate models than systems dominated by other components. While some have argued that CPU manufacturers should implement event counters for energy-related events [10], the current CPU performance counters do give some insight into power consumption. Similar high-level interfaces to low-level behavior do not exist for memory and disk, limiting the accuracy of this approach for systems where those components are dominant consumers of power. This problem is likely to increase in the future, since CPUs' percentage of overall system power is decreasing [1].

In general, across the wide variety of systems tested, the performance-counter-based power model we proposed met our accuracy requirements on all benchmarks and was general enough to be easily portable across systems. For some systems, simpler approaches may be just as good. The Mantis modeling methodology allows an entire family of models to easily be developed, and their trade-offs evaluated, for a wide range of computer systems.

7 Conclusions and Future Work

The results of this study suggest some future research directions. First, the models investigated in this work were very simple, not only in the number of inputs sampled, but also in the complexity of the equations used to obtain power predictions from utilization metrics. As Section 5 showed, these simple, mostly linear models may not adequately express the behavior of some systems, multicores in particular. Combining the empirical power model for CPU power with the information provided by disk utilization and CPU performance counters is one obvious extension. Furthermore, the models we investigated assume that the CPU, memory, and disk are the main consumers of dynamic power. Systems with other components, such as graphics processors or power-aware networking equipment, will require adjustments to the calibration suite. Furthermore, future power optimizations are likely to pose modeling challenges: in particular, the dynamic power consumption of the cooling system and aggressive power-management policies in individual components would have to be visible to the OS and incorporated in the model. Finally, as Section 5 showed, power models are less accurate for machines whose dynamic power consumption is not CPU-dominated. Since the CPU is likely to be a less dominant component in the future [1], it is important to understand how to develop accurate power models for other components. Part of the solution may be to offer high-level interfaces to detailed metrics for these components, analogous to CPU performance counters and their interface libraries.

In conclusion, we examined five different full-system power models generated by correlating AC power measurements with software utilization metrics. These models were evaluated on several computer systems with widely varying components and power footprints, identifying models that are both highly accurate and highly portable. This evaluation demonstrates the trade-off between simplicity and accuracy, and it also shows the limitations of previously proposed models based solely on OS-reported component utilization for systems where the CPU is either aggressively power-managed or is not the dominant consumer of dynamic power. Given hardware trends in exactly these directions, these models will be increasingly inadequate in the future, and models that use both OS-reported component utilizations and CPU performance counters will be increasingly necessary for accurate power prediction.

8 Acknowledgments

We would like to thank Dimitris Economou, Brian Zahnstecher, and Malena Mesarina for their help with the physical instrumentation of the Itanium server. Jacob Leverich helped provide the measurement infrastructure for the Xeon server and was indispensable in building the mobile fileserver. Finally, we are grateful to Stephane Eranian for his development of the perfmon library and his availability to answer questions.

References

[1] BARROSO, L. A., AND HÖLZLE, U. The case for energy-proportional computing. *IEEE Computer 40*, 12 (Dec. 2007), 33–37.

[2] BELLOSA, F. The benefits of event-driven energy accounting in power-sensitive systems. In *Proceedings of the ACM SIGOPS European Workshop* (June 2000).

[3] CONTRERAS, G., AND MARTONOSI, M. Power prediction for Intel XScale® processors using performance monitoring unit events. In *Proceedings of the International Symposium on Low-Power Electronics and Design (ISLPED)* (Aug. 2005).

[4] ECONOMOU, D., RIVOIRE, S., ET AL. Full-system power analysis and modeling for server environments. In *Proceedings of the Workshop on Modeling, Benchmarking, and Simulation (MoBS)* (June 2006).

[5] FAN, X., WEBER, W.-D., AND BARROSO, L. A. Power provisioning for a warehouse-sized computer. In *Proceedings of the International Symposium on Computer Architecture (ISCA)* (June 2007).

[6] HEATH, T., DINIZ, B., ET AL. Energy conservation in heterogeneous server clusters. In *Proceedings of the Symposium on Principles and Practice of Parallel Programming (PPoPP)* (June 2005).

[7] Intel® Core™ 2 Duo mobile processor product brief. Online. `http://www.intel.com/products/processor/core2duo/mobile_prod_brief.htm`.

[8] ISCI, C., AND MARTONOSI, M. Runtime power monitoring in high-end processors: Methodology and empirical data. In *Proceedings of the International Symposium on Microarchitecture (MICRO-36)* (Dec. 2003).

[9] JOSEPH, R., AND MARTONOSI, M. Run-time power estimation in high performance microprocessors. In *Proceedings of the International Symposium on Low-Power Electronics and Design (ISLPED)* (Aug. 2001).

[10] KADAYIF, I., CHINODA, T., ET AL. vEC: Virtual energy counters. In *Proceedings of the ACM SIGPLAN/SIGSOFT Workshop on Program Analysis for Software Tools and Engineering (PASTE)* (June 2001).

[11] LI, T., AND JOHN, L. K. Run-time modeling and estimation of operating system power consumption. In *Proceedings of the ACM SIGMETRICS International Conference on Measurement and Modeling of Computer Systems* (June 2003).

[12] MCCALPIN, J. D. STREAM: Sustainable memory bandwidth in high performance computers. Online. `http://www.cs.virginia.edu/stream/`.

[13] MOORE, J. COD: Cluster-on-demand. Online, 2005. `http://issg.cs.duke.edu/cod/`.

[14] NEDEVSCHI, S., POPA, L., ET AL. Reducing network energy consumption via sleeping and rate-adaptation. In *Proceedings of the USENIX Symposium on Networked System Design and Implementation (NSDI)* (Apr. 2008).

[15] NYBERG, C., AND KOESTER, C. Ordinal Technology – Nsort home page. Online, 2007. `http://www.ordinal.com`.

[16] PINHEIRO, E., BIANCHINI, R., ET AL. Load balancing and unbalancing for power and performance in cluster-based systems. In *Proceedings of the Workshop on Compilers and Operating Systems for Low Power (COLP)* (2001).

[17] RANGANATHAN, P., AND LEECH, P. Simulating complex enterprise workloads using utilization traces. In *Proceedings of the Workshop on Computer Architecture Evaluation using Commercial Workloads (CAECW)* (Feb. 2007).

[18] RANGANATHAN, P., LEECH, P., ET AL. Ensemble-level power management for dense blade servers. In *Proceedings of the Annual International Symposium on Computer Architecture (ISCA)* (June 2006).

[19] RIVOIRE, S. *Models and Metrics for Energy-Efficient Computer Systems*. PhD thesis, Stanford University, Stanford, California, 2008.

[20] RIVOIRE, S., SHAH, M. A., ET AL. JouleSort: A balanced energy-efficiency benchmark. In *SIGMOD Conference on Management of Data* (June 2007).