Statistical Profiling-based Techniques for Effective Provisioning of Power Infrastructure in Consolidated Data Centers

Sriram Govindan, Jeonghwan Choi, Bhuvan Urgaonkar, Anand Sivasubramaniam, and Andrea Baldini†
{sgovinda,jechoi,bhuvan,anand}@cse.psu.edu
Department of Computer Science and Engineering
The Pennsylvania State University, University Park 16802, PA

abaldini@cisco.com†
Cisco Systems, Inc.
170 West Tasman Drive, San Jose 95134, CA

Abstract

Emerging energy-aware initiatives (such as billing of power usage based on *de-coupling* between electricity sales and utility profits/fixed-cost recovery) render current capacity planning practices based on *heavy over-provisioning of power infrastructure* unprofitable for data centers. We explore a combination of statistical multiplexing techniques (including *controlled under-provisioning and overbooking*) to improve the utilization of the power hierarchy in a data center. Our techniques are built upon a measurement-driven profiling and prediction technique to characterize key statistical properties of the power needs of hosted workloads and their aggregates. As a representative result from our evaluation on a prototype data center, by accurately identifying the worst-case needs of hosted workloads, our technique is able to safely operate 2.5 times as many servers running copies of the e-commerce benchmark TPC-W as allowed by the prevalent practice of using face-plate ratings. Exploiting statistical multiplexing among the power usage of these servers along with controlled under-provisioning by 10% based on tails of power profiles offers a further gain of 100% over face-plate provisioning. Reactive techniques implemented in the Xen VMM running on our servers dynamically modulate CPU DVFS-states to ensure power draw below safe limits despite aggressive provisioning. Finally, information captured in our profiles also provides ways of controlling application performance degradation despite the above under-provisioning: the 95^{th} percentile of TPC-W session response time only grew from 1.59 sec to 1.78 sec.

1 Introduction and Motivation

To accommodate modern resource-intensive high-performance applications, large-scale data centers have grown at a rapid pace in a variety of domains ranging from research labs and academic groups to industry. The fast-growing power consumption of these platforms is a major concern due to its implications on the cost and efficiency of these platforms as well as the well-being of our environment. By 2005, the energy required to power and cool data center equipment accounted for about 1.2% of total

U.S. electricity consumption according to a report released by the Lawrence Berkeley National Laboratory and sponsored by chip manufacturer AMD. Gartner, the IT research and advisory company, estimates that by 2010, about half of the Forbes Global 2000 companies will spend more on energy than on hardware such as servers [17]. Furthermore, Gartner estimates that the manufacture, use, and disposal of IT equipment, a large share of which results from data centers, accounts for 2% of global CO_2 emissions which is equivalent to the aviation industry.

The rapid rise in the number of data centers and the growing size of their hardware-base (especially the number of servers) are the primary causes of their increasing power needs. As an example, a New York Times article in June 2006 reported that Google had approximately 8,000 servers catering to about 70 Million Web pages in 2001, with the number growing to 100,000 by 2003. Their estimate put the total number of Google servers (spread over 25 data centers) to be around 450,000 at the time. Similarly, Microsoft's Internet services were being housed in around 200,000 servers with the number expected to hit 800,000 by 2011. While the power consumption per-unit of hardware has contributed a much smaller percentage to this growth, continuing miniaturization at multiple levels (ranging from chips, servers, racks, to room) within the data center has necessitated the procurement of higher capacity cooling systems to deal with the growing power densities.

These trends have severe implications on the total cost of operation (TCO)—deployment-time costs as well as a variety of recurring costs—of a data center. The impact on TCO due to higher bills paid to the electricity provider are easy to appreciate. At a rating of around 250 Watts for a server (which is on the lower end of peak power for a dual CPU system, with 1-2 GB memory, a disk, and a network card), a data center with 20K-40K servers consumes between 5-10 Megawatts just towards powering these servers (discounting cooling system costs.) Assuming a cost of 10c/KWH, this amounts to $4.38M-$8.76M annually expended towards keeping these servers constantly powered for a single such data center. *Higher power consumption, however, also hurts the TCO in other less obvious ways.*

Existing practices for capacity planning of the power infrastructure within data centers employ significant degrees of over-provisioning at multiple levels of the spatial hierarchy, ranging from the power supplies within servers [24], Power Distribution Units (PDUs) supplying power to servers, storage equipment, etc., to even higher-level Uninterrupted Power Supply (UPS) sub-stations [15]. This over-provisioning is done to ensure uninterrupted and reliable operation even during episodes of excessive power draw as well as to accommodate future upgrades/additions to the computational/storage/networking equipment pool in the data center.

Such over-provisioning can prove unprofitable to the data center in two primary ways. The first and more significant reason emerges from ongoing efforts to promote energy-efficient operation and discourage wasteful over-provisioning of power supply via novel billing mechanisms. Currently, electricity providers make more money by selling more electricity and they make less money by helping their customers use less energy. Consequently, this disincentive impairs their willingness and ability to promote energy efficiency, despite its benefits to consumers' bills, electrical reliability, national security and the environment. This realization has led to proposals to remove the disincentives via "de-coupling," which means implementing a regulatory rate policy that breaks the link between electricity sales, on the one hand, and utility profits and fixed-cost recovery, on the other [19]. According to a USNews article in May 2008, while California pioneered de-coupling as early as the 1970s, Connecticut, Idaho, New York, and Vermont had also chosen de-coupling by 2007, and a dozen other states now are considering it [23]. Consequently, in the near future, a data center operating significantly below its provisioned power capacity can expect to pay higher recurring electricity costs. A

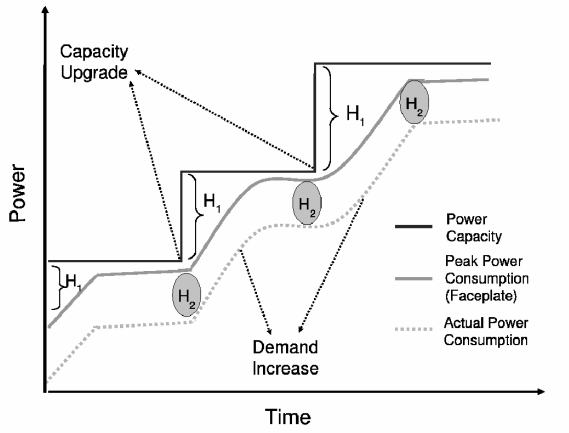


Figure 1: Illustration of the evolution of power capacity and demand in a hypothetical data center. Also shown is the evolution of provisioned capacity based on a prevalent practice such as using the face-plate ratings of devices.

second, perhaps comparatively minor, concern arises due to the excess costs expended towards procuring power infrastructure with higher capacity (including a larger number of power supply elements) than needed.

Decisions related to the provisioning of power infrastructure must be made not only at installation time but on a recurring basis to cope with upgrades. Figure 1 illustrates the evolution of the power demand and capacity in a data center. As shown, there are two "head-rooms" between power demand and capacity. The first head-room H_1 is intended to ensure the data center can accommodate foreseeable additions/upgrades (as shown by the curve labeled "Peak Power Consumption (Faceplate)" in Figure 1) to its hardware base. The second head-room H_2, our focus in this research, results due to current capacity planning practices that significantly over-estimate the power needs of the data center. A number of recent studies on power usage in data centers provide evidence of such over-provisioning at various power elements [15, 16, 24]. Building upon these insights, we plan to carefully understand the power usage of data center workloads in order to develop a provisioning technique that addresses the headroom H_2.

While provisioning closer to demand reduces both installation/upgrade as well as recurring costs, it does so at the risk of increased episodes of degraded performance/availability. This degradation can occur due to one or more of the following: (i) a subset of the hardware may simply not get powered up due to insufficient power supply (as happened with an ill-provisioned $2.3 Million Dell cluster at the University at Buffalo, where two-thirds of machines could not be powered on till a $20,000 electrical system upgrade was undertaken [9]), (ii) one or more fuses give way during an episode of surge in power drawn disrupting the operation of applications hosted on associated servers, and (iii) the thermal system, faced with constrained power supply, triggers shut/slow down of some devices. Any improvements in power provisioning must carefully trade-off the resulting cost savings against such performance degradation. Additionally, to realize such improvements, a data center must employ mechanisms that prevent (make statistically negligible) episodes of types (i)-(iii). In this paper, we develop a system that effectively provisions power while addressing these concerns.

In general terms, it is our contention that understanding the power usage behavior of hosted applications and employing techniques that are aware of these idiosyncrasies can allow a data center to make more informed provisioning decisions compared to existing techniques. To validate these ideas,

we explore a combination of several complementary approaches.

Research Contributions. The contribution of our research is three-fold.

- **Automatic derivation of power needs of aggregates.** We develop a profiling technique to derive statistical descriptions (called power profiles) of the power usage of hosted applications. We then develop prediction techniques that describe the power usage at different levels in the hierarchy based on profiles of applications placed under these levels (e.g., the power usage of a PDU supplying power to servers hosting a given set of applications.) Finally, we identify the limits posed by fuses throughout the hierarchy in the form of *sustained power budgets* so power usage of aggregates can be compared against these in systematic and statistically meaningful ways.
- **Yield management inspired provisioning techniques.** We exploit a number of statistical properties of power usage that our profiles reveal to propose improved provisioning techniques. First, *controlled under-provisioning* based on the tails of power profiles of hosted workloads exploits rarely occurring peak power needs. Second, identification of *statistical multiplexing effects* among workloads consolidated under a power supply element is used to carefully *overbook* power supply capacity. Finally, evidence of self-similarity in the power usage of an important class of workloads suggests that these under-provisioning related gains are likely to result not just for PDUs that servers are connected to, but even at higher levels of aggregation. Although these techniques are inspired by literature on yield management as employed in such diverse domains as the airline industry [32], telephony and networking [18, 40, 4], and CPU/network/memory resources in servers/data centers [39, 41], a number of characteristics peculiar to power infrastructure present us with novel concerns.
- **Agile protective systems mechanisms.** Finally, to enable safe and performance-friendly operation despite the above techniques for aggressive provisioning of the power hierarchy, we develop agile systems mechanisms based on dynamic throttling of the CPU DVFS state of our servers. Our profiling and prediction techniques enable a systematic trade-off between the cost savings offered by our provisioning technique and the accompanying performance degradation.

We implement our techniques in a prototype data center with a state-of-the-art PDU supplying power to multiple servers. We modify the Xen VMM to implement our agile protective mechanisms based on dynamically throttling CPU DVFS-states. Using a variety of well-regarded benchmarks representative of data center applications, we conduct a detailed empirical evaluation to demonstrate the feasibility and utility of our provisioning approach.

As a representative result, by accurately identifying the worst-case power needs of hosted workloads, our technique is able to improve the number of servers running copies of the e-commerce benchmark TPC-W that can be safely connected to a PDU by 150% compared to the currently prevalent practice of using face-plate ratings. Exploiting statistical multiplexing among the power usage of these servers along with controlled under-provisioning (by 10%) based on tails of power profiles offered a further gain of 100%. Furthermore, evidence of self-similarity in the power usage of some workloads suggests that such gains can be expected even higher up in the power hierarchy. Reactive techniques implemented in the Xen VMM running on our servers dynamically modulated CPU DVFS-states to contain power draw within safe limits despite our aggressive provisioning. Finally, information yielded by our profiles also provided ways of controlling the performance degradation resulting from our under-provisioning. For the experiment above, the average response time of TPC-W sessions only grew by 4% from 298 msec to 310 msec; the 95^{th} percentile of grew from 1.59 sec to

1.78 sec, a degradation of only 11.65%.

Road-map. The rest of this paper is structured as follows. We provide background on power consumption in data centers in Section 2. We conduct an empirical study of power consumption in consolidated settings in Section 3. Based on lessons learnt from this study, we develop techniques for improved provisioning and usage of power infrastructure in Section 4 and address associated reliability concerns in Section 5. We present our prototype implementation in Section 6 and conduct an experimental evaluation of our techniques in Section 7. Finally, we discuss related work in Section 8 and conclude in Section 9.

2 Power Provisioning Overview

In this section, we provide necessary background on the power supply infrastructure and the hosting model assumed in our data center.

Power Supply Hierarchy. In a typical data center, a primary switch board distributes power among several *Uninterrupted Power Supply Sub-stations* (UPS; 1,000 KW) that, in turn, supply power to collections of *Power Distribution Units* (PDU; 200 KW.) A PDU is associated with a collection of server racks (up to 50.) Each rack has several chassis that host the individual servers. Power supply could be either at the server-level (as in rack-mounted systems) or at the chassis-level (as in blade servers.) Within all these components, fuses/circuit-breakers [1] are used to protect equipment from surges in the current drawn. We view this collection of power supply infrastructure as forming a *power supply hierarchy* within the data center, with the primary switch board at the top, the power supplies within servers (and storage/networking equipment) at the bottom, and UPS units, PDUs, etc. in between.

Power Budgets. Each fuse or circuit-breaker has a time-current characteristic curve: a point (s, l) on this curve specifies the maximum power draw s that the fuse can safely sustain over l *contiguous* time units. For simplicity, we use a single such point (S, L) as the *sustained power budget* for the corresponding level in the hierarchy. Sustained power budgets are defined over fairly small time periods—of the order of a few seconds or even milliseconds. A violation of this sustained power budget would mean a draw of S Watts or more was sustained over a contiguous period of L time units. While sustained power is closely related to the notion of *peak power* that is frequently used in literature [16, 28], the difference must be clearly understood. Peak power, as is typically defined, merely captures the maximum power consumed within an entity/level without capturing the time-scale over which this power usage sustains. Hence, it does not capture the limits posed by fuses well. For example, a device may have a high peak power consumption and still operate safely if this peak does not sustain long enough to exercise the limits associated with the corresponding fuse.

Virtualized Hosting Model. Our hosting model assumes a large cluster of high-end servers (say with dual processors and a few GB of memory) interconnected by a high bandwidth network for communication. Each server runs a virtual machine monitor (VMM) allowing multiple applications to be

[1] A circuit breaker is similar to a fuse in its function except that it could be reused after an episode of excessive current draw. We will simply use the term fuse for both henceforth.

consolidated within a single server if their resource needs can be accommodated by the server. In addition, many of these servers are also connected to a consolidated high capacity storage device/utility through a Storage Area Network which facilitates data sharing and migration of applications between servers without explicit movement of data.

3 Power Profiling and Prediction for Aggregates

In this section, we develop techniques to measure and characterize the power consumption of individual applications. Borrowing techniques from existing research, we also derive characterizations of their resource usage. Finally, we develop techniques to predict the power consumption at various levels of the spatial hierarchy when these applications are consolidated. Taken together, these measurements and techniques set the background for improvements in provisioning the power infrastructure that we explore in the following sections.

3.1 Empirical Derivation of Power and Resource Usage Profiles

Our approach for characterizing the power and resource usage of an application employs an offline profiling technique. The profiling technique involves running the application on an isolated server. By isolated, we mean that the server runs only the system services necessary for executing the application and no other applications are run on the server during the profiling process. Such isolation is necessary to minimize interference from unrelated tasks when determining the application's power and resource usage. [2] The application is then subjected to a realistic workload and a combination of hardware and software monitoring infrastructure is used to track its power and resource usage.

Profiling Power Consumption. We connect a multi-meter to the server used for our offline profiling and use it to measure the power consumption of the server once every t_p time units. We find it useful to convert the resulting time-series of (instantaneous) power consumption samples into a probability density function (PDF) that we call the application's *power profile*. Let $w_A^{I_p}$ be a random variable that represents the average power consumption of the application A over durations of I_p time units, where $I_p = k \cdot t_p$, (k is a positive integer.) Note that $w_A^{I_p}$ represents the average consumption over *any* consecutive interval of size I_p. It is estimated by shifting a time window of size I_p over the power time-series, and then constructing a PDF from these values. Figure 2 illustrates the process of converting a power usage time-series into a power profile. Notice that our technique takes each power sample to be the power consumption *throughout* the t_p time units preceding it. Clearly, the inaccuracies due to this assumption grow with t_p. Finally, as part of our profiling, we also record the idle power of the server running the applications (127-141 W for our servers.)

Profiling Resource Usage. We use measurement techniques similar to those existing in research [1, 30, 39] to record resource scheduling events of interest. By recording CPU scheduling/de-scheduling

[2] In practice, a distributed application with multiple components may require multiple servers to meet its resource needs. We only consider applications whose resource needs can be met by a single server. Our technique extends to applications requiring multiple servers by simply running the application on the appropriate number of servers and conducting measurements on each of them.

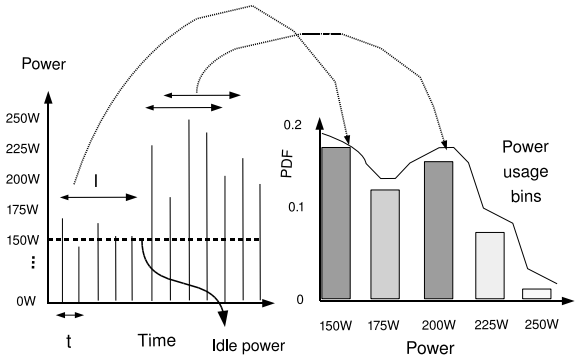


Figure 2: Illustration of the derivation of a power profile from a power consumption time-series for $t_p = t$ and $I_p = I$.

instants for the virtual machine running our application, we derive its CPU usage profile, an ON-OFF time-series of its CPU usage. Similarly, packet transmission/reception times and lengths yield its network bandwidth usage profile. We also record time-series of memory consumption and disk I/O requests made by the application. Similar to power measurements, we find it useful to construct resource usage PDFs from these profiles. Finally, we also record PDFs of application-specific performance metrics (e.g., response time, throughput.)

Discussion on Our Profiling Technique. The efficacy of our provisioning depends crucially on the credibility as well as the feasibility of our offline profiling.

- *On the feasibility of collecting profiles*: The workload used during profiling should be both realistic and representative of real-world workloads. There are a number of ways to ensure this, implying that offline profiling is not unreasonable to assume. While techniques for generating such workloads are orthogonal to our current research, we note that a number of different well-regarded workload-generation techniques exist, ranging from the use of synthetic workload generators to the use of well-known benchmarks, and from trace replay of actual workloads to running the application in a "live" setting. Any such technique suffices for our purpose as long as it realistically emulates real-world conditions. In fact, (with regard to running an application in a live setting) many data center applications do start in isolation. Consequently, profiling can be done during the early stages of application deployment, similar to that proposed in current research [39, 37]. Furthermore, workload patterns are often repetitive over some time granularity (such as daily cycles [20]), providing opportunities to incorporate increased confidence into gathered profiles by conducting multiple measurements.
- *Dealing with varying resource/power usage*: Implicit in the power/resource profiles described above is an assumption of stationarity of power/resource usage behavior. Executions of realistic applications are likely to exhibit "phases" across which their power and resource usage behavior change significantly. An example of this is the change in resource needs (and hence power consumption) of a Web server whose workload exhibits the well-known "time-of-day" variation [20]. Similarly, many scientific applications alternate between doing significant amounts of I/O (when reading in parameters from files or dumping results to them) and computation. Clearly, the utility

of our power profiles depends on effectively determining such phases. Power and resource profiles could then be derived separately for every such phase. Enhancing our techniques to deal with these issues is part of our future work. In this paper, we limit ourselves to a single profile per-application, except in Section 7.4, where we explore a simple technique to detect a significant change in a power profile.

- *Measurement infrastructure related considerations*: Note that due to our ability to only measure power usage at the granularity of the entire server (as opposed to measuring the power usage of constituent components such as CPU, disk, etc.), our measurements are only an approximation (in fact, an upper bound) of the power consumed by the application. By minimizing any other interfering activities during the offline profiling, we attempt to keep this gap small. A related issue concerns the parameters t_p and I_p involved in the definition of the variable $\underline{w}^{I_p}$. We empirically study appropriate values for these in our recent research [7]. Finally, we restrict our attention to profiles with the CPU operating at the highest DVFS state. Results presenting profiles at different DVFS states may be found in [7].
- *On application modeling*: We do not concern ourselves with identifying relationships between application's performance metrics (such as response time) and resource usage. This is a well-studied area in itself [36, 3, 10, 14, 38]. We borrow from this literature whenever it is easily done. Generally, we make simplifying assumptions about these dependencies that we expect not to affect the nature of our findings.

3.2 Profiling Applications: Key Experimental Results

In this section, we profile a diverse set of applications to illustrate the process of deriving an application's power consumption behavior. We also present selected information about resource usage and performance. These experiments provide us a number of key insights into: (a) the relationship between an application's power consumption and its usage of various resources and (b) important statistical characteristics of the power usage behavior of these applications.

Dell PowerEdge SC1450 Features [12]	
Processor	Two(2) Intel(R) Xeon 64bit 3.4 GHz
Main Memory	2GB
L2 Cache	2MB
Hard Disk(2)	WD Caviar 40GB 7200rpm
Hard Disk Power	7W/1W (Active/Standby)
Network Interface	Dual embedded Intel Gigabit2 NICs
Power Supply	450Wx1

Table 1: Specifications of the server used to host the applications that we profile.

Signametrics SM2040 Features [31]	
Digits of Resolution	6-1/2
Measurement Rates	0.2/sec - 1000/sec
Measurement Range (AC current)	2.5A
Interface	PCI

Table 2: Details of the multi-meter used in our profiling.

Our testbed consists of several Dell PowerEdge servers (details appear in Table 1.) We use one of these servers for running the applications that we profile. We connect a Signametrics SM2040 multi-meter (details appear in Table 2) in series with the power supply of this server. The multi-meter

sits on the PCI bus of another server which is solely used for logging purposes. This multi-meter is capable of recording power consumption as frequently as once every millisecond. Unless otherwise specified, throughout this section, we have t_p=I_p=2 msec. In Section 7, we employ a less accurate measurement facility within a PDU in our prototype data center to be able to simultaneously record power consumptions of multiple servers connected to it. We run the Xen VMM [2] on our servers with each application encapsulated within a Xen domain.

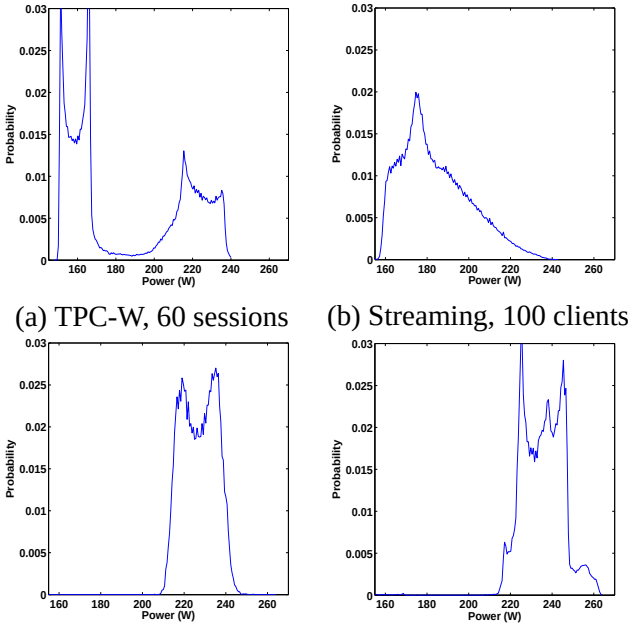
The server running the application is connected to the multi-meter while the remaining servers are used to generate the workload. We report our observations for the representative applications listed in Table 3. In our environment, the CPU is the largest contributor to power consumption, so we find it useful to classify these applications based on their CPU usage. Applications in the SPEC CPU2000 suite are *CPU-saturating*, in that they are ready to use the CPU at all times. The remaining applications alternate between using the CPU and being blocked (e.g., on I/O, synchronization activities, etc.) and their CPU utilization depends on the workload they are offered. We profile these *non-CPU-saturating* applications at different workload intensities. TPC-W is profiled with the number of simultaneous Web sessions varying from 10 to 100, in increments of 10. For experiments involving TPC-W, we represent the workload intensity as TPC-W(x) where "x" is the number of simultaneous Web sessions. For experiments involving Streaming Media Server, 3Mbps is used as the streaming rate; Streaming(x) represents a workload of "x" clients. Finally, the workload intensity for SPECjbb2005 can be controlled using a tunable parameter dictating the number of "warehouses" it stores. We use a value of 6 for this parameter throughout this paper.

Applications	
TPC-W [33]	3-tiered NYU implementation of the TPC-W Transactional Web-based E-commerce benchmark
Streaming Media	Home-grown UDP streaming server, Streams MPEG-1 to specified no. of clients & data rate
SPECjbb2005 [35]	SPEC's 3-tiered client-server benchmark Emulates server-side Java applications
SPEC CPU2000 [34]	SPEC CPU2000 suite (Art, Bzip2, Mcf, Mesa)

Table 3: Salient properties of our applications. TPC-W, Streaming Media Server, and Specjbb2005 are non CPU-saturating, whereas applications in the SPEC CPU2000 suite are CPU-saturating.

We now present key results from our profiling study.

Temporal Variations in Power Usage. We find that all our applications exhibit temporal variations in their power usage to different degrees. Given that CPU consumes significantly more power than I/O devices in our environment, not surprisingly, power profiles for non CPU-saturating application (Figures 3 (a)) are found to exhibit higher variance than CPU-saturating application (Figures 3 (b).) Specifically, in the profiles reported in Figure 3, the variance of the TPC-W and Streaming profiles were 92 and 84 compared with only 47 and 59 for Bzip2 and Mcf, respectively. The CPU usage of a non CPU-saturating application exhibits an ON-OFF behavior, corresponding to the application being in running and blocked states, respectively. When such an application blocks, its power consumption corresponds to the server's idle power. This ON-OFF CPU usage contributes to the higher variance in its power consumption. Intuitively, we expect that on being consolidated under common power elements, applications with higher variance in their power usage would yield larger reductions (over worst-case provisioning) in required power capacity via statistical multiplexing effects. The exact extent of these savings would depend on the particular set of applications being consolidated together.



(a) TPC-W, 60 sessions (b) Streaming, 100 clients

(c) Bzip2 (d) Mcf

Figure 3: Power distributions of TPC-W(60), Streaming(100), Bzip2, and Mcf compared.

Application	Power usage percentile (W)				
	100^{th}	99^{th}	95^{th}	90^{th}	Avg.
TPC-W(60)	260.4	236.4	233.2	229.2	185.5
Streaming(100)	242.4	227.4	214.8	208.2	184.1
Bzip2	252.6	242.4	237.2	235.1	224.9

Table 4: Salient aspects of the power profiles of TPC-W, Streaming, and Bzip2 sampled at 2 msec granularity.

Tails of Power Profiles. The nature of the tail of a resource requirement distribution crucially affects savings that under-provisioning (that is, provisioning less than what the worst-case needs suggest) can yield. In Table 4, we present the 99^{th}, 95^{th}, and 90^{th} percentiles of the power profiles of TPC-W(60), Streaming(100), and Bzip2 along with their peak and average values. We make two useful observations. First, for all the applications, the worst-case power needs (in the range 240-260 W) are significantly less than the power supply provisioned within our server (450 W, recall Table 1.) Second, the 99^{th} and 95^{th} percentile needs are lower than the worst case by up to 10%, while the 90^{th} percentile is lower by up to 15%. Together these results suggest that controlled under-provisioning based on power profile tails can potentially bring about capacity and cost savings.

Self-similarity in Power Usage. A final statistical feature worth investigating in our profiles is the presence (and extent) of self-similarity [26]. Due to the hierarchical nature of the power infrastructure (recall Section 2), the presence of self-similarity has interesting implications on capacity provisioning at higher layers of aggregation (PDU, UPS, etc.) The well-known Hurst parameter (H) is one way to quantify the self-similarity exhibited by a process. It lies in [0.5, 1.0] with higher values representing larger degrees of self-similarity. We calculate the Hurst parameter for the power time-series of our applications. We find the Hurst parameter to be 0.86, 0.83, 0.76, and 0.52 for TPC-W(60),

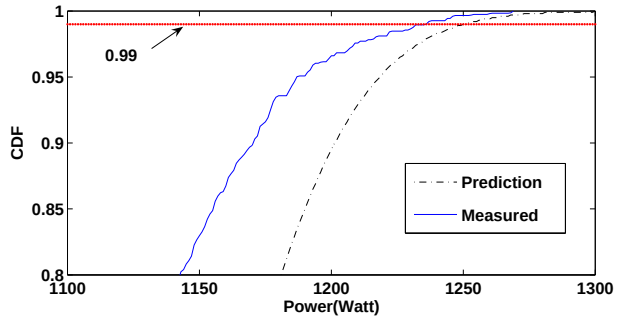


Figure 4: Comparison of measured and sustained power consumption (L=5 sec) of a PDU connected to 7 Servers (each running TPC-W(60).)

SPECjbb2005, Bzip2 and Streaming(100), respectively. Due to the hierarchical nature of the power infrastructure (recall Section 2), the presence of self-similarity has interesting implications on capacity provisioning at higher layers of aggregation (PDU, UPS, etc.) Specifically, applications with (i) long tails in their power profiles *and* (ii) high self-similarity in their power time series, are likely to retain these characteristics (i.e., long tails/burstiness) even at higher levels of aggregation. In particular, since TPC-W(60) has a long tail (refer Table 4) and a high Hurst parameter, we expect the aggregate power series of multiple TPC-W(60) server instances to also exhibit burstiness. SPECjbb2005, that exhibits a high Hurst parameter along with low burstiness, presents a contrasting case: we expect power elements consolidating copies of this application to experience power usage patterns with low burstiness. We validate these intuitions in Section 3.3 where we study power usage of such aggregates.

3.3 Prediction Techniques for Aggregates

Crucial to provisioning levels in the hierarchy higher than the server (PDU, UPS, etc.) are ways to combine the power profiles of applications running on the servers beneath this level to predict their aggregate power usage. While predicting the average and even the peak of such an aggregate is fairly straightforward, doing the same for sustained power (recall the definition in Section 2) is non-trivial. We employ our recent research which combines power and resource usage profiles of individual applications and predicts the behavior of sustained power consumed at various levels (server, PDU, and higher) when these are consolidated. A representative result is presented in Figure 4 and Table 5. As shown, for a PDU connected to 7 servers, each consolidating TPC-W(60), our technique predicts within reasonable error margins (1-5%). For a detailed explanation of our prediction technique, please refer to our Technical Report [7].

As suggested in Section 3.2, we observe that the sustained power consumption of this collection of servers, each running an instance of the relatively bursty TPC-W(60) application, also exhibits a longer tail (e.g., compare the 100^{th} and 90^{th} percentiles reported in Table 5) than those for aggregates of (i) the less bursty SPECjbb2005 application and (ii) the less self-similar Streaming Server (see [7] for more details.)

Power percentile	Measured Sustained power (W)	Predicted sustained power (W)	Error (%)
80	1143	1181	3.2
90	1171	1201	2.4
99	1236	1250	1.1
100	1269	1300	2.4

Table 5: Efficacy of our sustained power prediction on a PDU consolidating 7 Servers each running TPC-W(60). We compare the tail of the measured power with our predicted power.

4 Improved Provisioning of Power

In this section, we propose techniques that utilize the profiling and prediction techniques developed in Section 3 to better provision the power hierarchy in a data center. While doing this, the data center must strike a balance between the cost savings and the performance degradation likely to result due to the protective mechanisms that enable safe operation during episodes of power draw in excess of provisioned capacity.

4.1 Under-provisioning Based on Power Profile Tail

Sections 3.2 and 3.3 reinforce recent results suggesting that provisioning based on the face-plate ratings of servers severely under-utilizes the power infrastructure [24, 15]. In fact, these results suggest that a data center can even go a step further—given the extent of burstiness present in the power usage of many commonly hosted applications, we can realize further improvements by provisioning less conservatively than for the worse-case. This has two complementary implications at each level of the power hierarchy: (i) its power supply can be replaced with one with lower capacity (and cost) and/or (ii) it can supply power to a larger overall set of devices connected in the levels beneath it. (Although we will focus on (ii), the gains in (i) are easily understood as well.)

For ease of exposition, let us assume that all sustained power budgets in the following discussion are defined over a unit time period - the second element of the sustained power budget pair will therefore be omitted. It is easy to generalize this discussion to budgets defined over arbitrary time periods. Let us denote by B the sustained power budget associated with a power supply element E. Let n elements drawing power from E be denoted $e_1, \ldots, e_n$, and their (predicted) sustained power profiles be denoted $u_1, \ldots, u_n$. [3] Finally, let u^p denote the p^{th} percentile of the distribution u. Under-provisioning the capacity at element E implies ensuring the following condition:

$$\sum_{i=1}^{n} u_i^{100-p_i} \leq B; \quad \forall i : p_i > 0. \tag{1}$$

This should be compared with provisioning the power capacity of this element for the worst-case needs:

$$\sum_{i=1}^{n} u_i^{100} \leq B. \tag{2}$$

The degree of under-provisioning p_i for element e_i should be chosen based on a desirable trade-off between the cost savings accrued from under-provisioning and the performance degradation that occurs.

[3] This notation is general enough to capture under-provisioning at any level. For example, if E denotes a server (with its power supply being the candidate for under-provisioning), the elements e_i are applications consolidated on it. The specific level for which we evaluate our techniques considers a PDU as the element E supplying power to servers denoted by e_i.

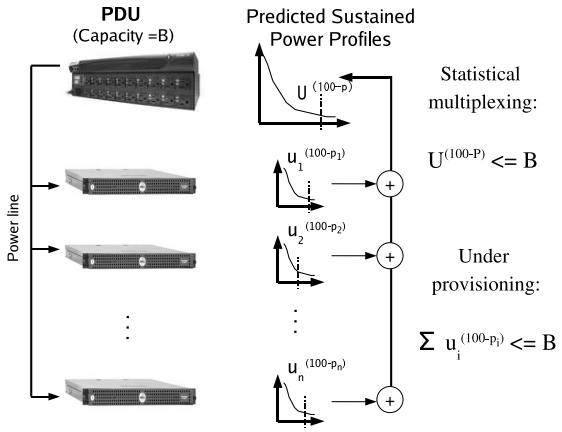


Figure 5: Illustration of our techniques based on under-provisioning and statistical multiplexing/over-booking as applied to a PDU supplying power to a collection of servers.

The gain offered by the provisioning as represented by formula (1) over worst-case provisioning is $\sum_{i=1}^{n}(u_i^{100} - u_i^{100-p_i})$. Clearly, applications with more bursty power profiles would yield higher gains at the PDU level. Furthermore, as argued in Section 3.2, elements consolidating bursty workloads underneath them are also likely to experience bursty power usage, implying gains even at these higher levels.

4.2 Exploiting Statistical Multiplexing

If the elements consolidated at a given level exhibit power usage patterns that complement each other temporally, then statistical multiplexing gains become worth exploiting. In simple terms, *provisioning for the tail of aggregates can be more gainful than provisioning for the sum of individual tails* (as was done in the under-provisioning technique above.) In Section 3.2, we saw evidence of appreciable temporal variations for a subset of our applications. Adding to the terminology introduced above the symbol $\mathcal{U}$ for the sustained power profile at element E, we can enhance our provisioning technique as follows:

$$\mathcal{U}^{100-p} \leq B; \quad p > 0. \tag{3}$$

Rather than under-provisioning the "share" of each element e_i independently as in (1), this technique does so for the *aggregated needs* of all these elements. A key point to note here is that under-provisioning and statistical multiplexing are *not mutually exclusive but complementary*—the aggregation $\mathcal{U}$ representing multiplexing of underlying power usages is being under-provisioned. The degree of under-provisioning p should be chosen based on the following considerations. First, it should be possible to distribute it into individual degrees of under-provisioning (p_i for element e_i) that provide desirable trade-offs between cost savings and performance. Second, (as mentioned in the last technique) mechanisms should be present at (and below) the level of elements e_i in the hierarchy to enforce the power limits corresponding to these degrees of under-provisioning. We will address these issues in Section 5.

4.3 Controlled Over-booking of Power

A final enhancement to our provisioning technique incorporates *over-booking* of power capacity at E. Intuitively, if the power needs of the aggregate consolidated below element E are substantially lower than the capacity, then a small degree of over-booking can further improve the gains offered by statistical multiplexing. We incorporate an over-booking factor $\mathcal{O}$ as follows to achieve this:

$$\mathcal{U}^{100-p} \leq B \cdot (1 + \mathcal{O}); \quad p, \mathcal{O} > 0. \tag{4}$$

Figure 5 summarizes all of these techniques for a PDU supplying power to a group of servers.

5 Reliability and Performance Concerns

We consider, in turn, concerns of reliability and performance that must be addressed to gainfully utilize the provisioning techniques developed so far.

5.1 Enforcement of Power Budgets

Our techniques result in (or increase) the likelihood of episodes where the power needs at one or more levels within the hierarchy exceed its capacity. The over-provisioning based practices prevalent currently render such events practically impossible. Unless remedial actions are taken during such an occurrence, extremely undesirable outcomes (e.g., a subset of the hardware becoming unavailable, thermal exigencies that could affect the reliability of hardware, among others) could result. Realizing any meaningful usage/cost gains for the power infrastructure may require setting the provisioning parameters (e.g., p and O introduced in the previous section) throughout the hierarchy high enough to make the likelihood of budget violations non-negligible. Furthermore, unpredictable/hard-to-predict workload changes (such as an overload experienced by an e-commerce site [22]) may also render budget violations more likely than predicted by profiling based on prior workload patterns. These concerns necessitate mechanisms within a data center that can *completely avert such episodes.* Fortunately, as demonstrated by several recent research efforts, such mechanisms are realizable (more details in Section 8.)

We rely on the ability of the consumers of power (e.g., servers in our work) to operate at multiple "power states" (e.g., CPU DVFS state) that allow trade-offs between power consumption and resource capacity. We employ reactive techniques based on watermarks within our power hierarchy that utilize dynamic transitions to lower power states to avert power budget violations. Conceptually, when the watermark for an element in the hierarchy is exceeded, the data center triggers throttling of the appropriate subset of its hardware. A watermark for an element with a sustained power budget (s, l) is a 2-tuple $(s_w \leq s, l_w \leq l)$ and has the following operational meaning: Upon observing a sustained power draw at a rate of s_w units or more for l_w time units, an element should initiate the throttling of the consumers under it in the hierarchy. As we will discuss in Sections 7.2 and 8, l_w chosen to allow enough reaction time for the throttling to kick in, can provide the desired reliable operation. The choice of a watermark has to strike the following balance: Higher values of l_w reduce the number of invocations of throttling while resulting in poorer application performance upon these invocations. We can borrow from the findings of existing research, particularly [44], on this front. In our implementation and evaluation, we use a simple, statically-chosen watermark (see Sections 6 and 7.2.)

Application	Power usage percentile (W)		
	100^{th}	90^{th}	Avg.
TPC-W(60)	209	199	164.3
TPC-W(20)	183	152	150
Streaming(100)	183	159	152.1
Specjbb2005	219	219	217.0

Table 6: Salient aspects of the power profiles of TPC-W, Streaming Server, and Specjbb2005 collected by running these applications on servers connected to our PDU. Power is sampled at 1 second granularity.

5.2 Performance Concerns

Like any system employing under-provisioning and/or statistical multiplexing of a resource among competing consumers, our data center must contend with the accompanying degradation in resource availability and the resulting performance deterioration experienced by hosted applications. The provisioning parameters (p and O for each level within the power hierarchy) and the nature/efficacy of our throttling protective mechanisms determine how the performance of hosted applications is affected.

As in Section 5.1, our data center can gainfully borrow ideas from several bodies of research on understanding the relationship between application performance and different degrees of resource availability (application modeling), including work by the authors [6, 38, 39]. Information regarding the following aspects of application behavior captured by our profiling technique could be utilized by such models to achieve desired trade-offs between the cost gains and performance degradation: (i) resource capacity and performance offered to applications at different power states and (ii) power profiles with the equipment operating at different power states. While we restrict ourselves to a simple DVFS modulation scheme (described in evaluated in Section 7.2) in this paper, we have developed more sophisticated schemes for power/performance trade-off and control in related research [5].

6 Implementation Considerations

Infrastructure. Our experimental testbed consist of a 20 Amps PDU from Raritan Inc. [29] that can supply power to up to 20 servers. The PDU provides a software interface to read the power consumption of each server connected to it as well as the power consumption of the entire PDU. The granularity of the power measurement is 1 second and accuracy is 0.1 Amp. For our experiments, we vary the number of servers connected to the PDU. Note that the measurement capability offered by the PDU is lower fidelity than the multimeter used in Section 3 (every msec with accuracy within 10^{-6} Amp.) We repeat the profiling experiments described in Section 3 using the PDU and report important power consumption characteristics in Table 6. A dedicated set of servers (other than the ones that were connected to the PDU) were used for generating the workloads. Each server hosting an application runs the Xen VMM 3.1.0 (modified as described below) with each applications encapsulated within a separate domain. While our techniques apply to scenarios where multiple applications are consolidated on a single server (see [7] for our related efforts exploring profiling and prediction in such settings), we restrict ourselves to hosting one application per server. To enable live migration [8] of these VMs across servers, we place all the VM disk images in a NFS partition that is exported to all the servers connected to the PDU.

Our servers have a rich set of power states including 4 DVFS and 8 Clock Modulation states (refer

Table 1.) We switch off the default Linux daemon within Xen's administrative *Domain0* that dynamically varies the power state based on processor utilization. We write custom drivers for changing the power state of our servers. We use the IA32_PERF_CTL and IA32_CLOCK_MODULATION MSR registers to change the DVFS and clock modulation states, respectively. Since the vanilla Xen VMM traps and nullifies all writes to MSR registers (*wrmsr* operations), we modify it to enable writes to these registers.

Watermark-based Budget Enforcement. We briefly discuss the implementation of our technique based on a watermark (s_w, l_w) for enforcing a sustained power budget (s, l). We dedicate a server other than those hosting the applications or generating workloads to initiate reactive throttling and call it the *watermark-based enforcer*. The watermark-based enforcer periodically (once every 1 sec) monitors the power consumption of the PDU and inspects all the power samples collected over the last l_w time units. If all these values exceed s_w, it sends throttling commands to all the servers connected to the PDU using RPCs that specify their new power states. In Section 7, we will discuss in detail how the watermark-based enforcer selects appropriate throttling states for the servers.

7 Experimental Evaluation

7.1 Improvements in consolidation

In this section, we compare prevalent provisioning techniques used in data centers with the techniques developed in Section 4. We restrict our investigation to capacity/cost improvements in the number of servers that can be connected to a PDU and safely operated; similar improvements are worth exploring at other levels in the power hierarchy. For all our experiments we assume the sustained power budget for the PDU to be (1200W, 5 sec). We compare the following provisioning techniques.

Face-plate Provisioning (FP). Face-plate value is the capacity rating of a server specified for its power supply. For our servers the face-plate value is 450W. Using FP, we can connect 2 servers to our PDU.

Vendor Calculator-based Provisioning (VP). Server vendors (including IBM, HP, Sun, and Dell), in an attempt to help data centers administrators, provide calculators for estimating the peak power needs of their servers. Such a calculator takes as input the configuration of a server (number and type of processors, memory cards, etc.) and expected workload intensity (rough descriptions of CPU, I/O intensity, etc.) and outputs its power needs. The calculator provided by the vendor of our server [13] (for average load specification) estimates its power requirement to be 385W. Therefore using this provisioning technique, we would connect 3 servers to our PDU.

Profiling-guided Provisioning (UP and SP). The last two prevalent provisioning techniques are based solely on worst-case estimates of server power needs. In contrast, the techniques developed in Section 4 incorporate application-specific power needs. Let us denote by $UP(p_i)$ our under-provisioning based technique (recall (1)) and by $SP(p)$ the statistical multiplexing based technique (recall (3).)

Technique	Servers running instances of TPC-W(60)	
	No. Servers	% Improvement
$UP(100)$	5	66
$SP(100)$	6	100
$UP(90)$	6	100
$SP(90)$	7	133

Table 7: The number of servers (each running an instance of TPC-W(60)) connected to a 1200W PDU by different provisioning techniques. Percentage improvements reported are over VP.

Technique	Servers running instances of SPECjbb2005	
	No. Servers	% Improvement
$UP(100)$	5	66
$SP(100)$	5	66
$UP(90)$	5	66
$SP(90)$	5	66

Table 8: The number of servers (each running an instance of SPECjbb2005) connected to a 1200W PDU by different provisioning techniques. Percentage improvements reported are over VP.

In theory, $UP(100)$ and $SP(100)$ should coincide. However, due to extremely small probabilities (smaller than 10^{-7}) being rounded off to 0 in our implementation of sustained power prediction, we observe a difference between these quantities. In fact, these differences add up to slightly more than 150W at a PDU connected to 7 servers, each running an instance of TPC-W(60).

Tables 7-9 present improvements yielded by UP and SP in the number of servers hosting a diverse mix of applications that can be connected to our PDU with either: (i) no under-provisioning or (ii) under-provisioning of 10%. (Improvements resulting from the over-booking parameter $\mathcal{O}$ (recall 4) would be qualitatively similar, and we do not investigate them here.) Whereas the worst-case sustained power consumption of SPECjbb2005 and TPC-W are close to each other (220W and 210W respectively as shown in Table 6), due to the longer tail in its profile, higher gains result in an environment with servers hosting TPC-W like workloads. In fact, for an environment with SPECjbb2005-like applications, while provisioning based on $UP(100)$ (i.e., worst-case needs) provides 66% improvement over Vendor Calculator-based provisioning and 150% improvement over face-plate based provisioning, no further improvements result from under-provisioning. we are able to achieve higher consolidation for servers running TPC-W(60) as illustrated in Table 7 whereas no improvement could be seen for servers running SPECjbb2005 as shown in Table 8. Table 9 illustrate the efficacy of our provisioning technique for a set of servers running different kinds of applications. Gains offered by our provisioning techniques are thus closely dependent on the power usage characteristics of the hosted workloads.

Finally, we explore a variety of application mixes that lie between the previous two extremes. We present a subset of these in Table 9 and find gains ranging from 33% to 133% over VP.

7.2 Sustained Budget Enforcement

We evaluate the efficacy of our watermark-based budget enforcement technique developed in Sections 5.1 and 6. For our PDU's sustained power budget of (1200W, 5 sec), we choose (1200W, 3 sec) as the watermark.

We evaluate the efficacy of budget enforcement for an increasing number of servers—starting at 6

Technique	No. servers hosting each type of app.	% Improvement
$UP(100)$	3 x TPC-W, 1 x SPECjbb, 1 x SM	66%
$SP(100)$	3 x TPC-W, 1 x SPECjbb, 2 x SM	100%
$UP(90)$	3 x TPC-W, 1 x SPECJBB, 2 x SM	100%
$SP(90)$	3 x TPC-W, 1 x SPECJBB, 3 x SM	133%
$UP(100)$	2 x TPC-W, 2 x SPECjbb	33%
$SP(100)$	2 x TPC-W + 2 x SPECjbb, 2 x SM	100%
$UP(90)$	2 x TPC-W, 2 x SPECjbb, 2 x SM	100%
$SP(80)$	2 x TPC-W, 2 x SPECjbb, 3 x SM	133%

Table 9: The number of servers connected to a 1200W PDU by different provisioning techniques. Percentage improvements reported are over VP. Each server runs an instance of one of the following: TPC-W(60), SPECjbb2005, and Streaming(100), shortened to TPC-W, SPECjbb, and SM, respectively.

Power state	Predicted Peak of Sustained Power			
(DVFS, Clk. Mod.)	6 servers	7 servers	8 servers	9 servers
$(3.2Ghz, 100\%)$	**1191.0 W**	1300.0 W	1481.0 W	1672.0 W
$(2.8Ghz, 100\%)$	967.6 W	**1138.6 W**	1308.2 W	1478.2 W
$(2.8Ghz, 50\%)$	861.7 W	1011.7 W	**1162.7 W**	1313.6 W

Table 10: Power consumption of a server running TPC-W(60) when operating at three different power states. Bold power values indicate that the corresponding power state is chosen for throttling by the watermark-based enforcer. *Legend:* Clk. Mod.= Clock Modulation state.

and going up to 9—connected to a PDU. Each server runs an instance of TPC-W(60). The watermark-based enforcer described in Section 6 sends throttling commands to the servers upon observing three consecutive power readings above 1200W at the PDU (recall that the sampling interval is 1 sec.) Upon observing such an episode, the watermark enforcer must choose suitable power states for throttling the servers so that the sustained budget remains un-violated. This is achieved using a combination of our sustained power prediction technique and information gathered during offline profiling. In Table 10, we record the peak of the sustained power consumption at the PDU for varying number of servers connected to it and made to operate at different power states. For each of these server aggregates, we choose the highest power state, for which the peak of sustained power consumption is less than the PDU's budget. This chosen power state (highlighted in Table 10 for server aggregates of different sizes) is therefore guaranteed to bring the system within the capacity limits. As we can see from the table, there is no such power state if 9 servers, each running TPC-W(60), were connected to our PDU. That is, even if we operate at the lowest possible power state, [4] our technique can not prevent violations of the PDU budget. Throttling is done for a period of 2 seconds (which is the difference between the time constants of the watermark and the sustained power budget) after which the servers

[4]Actually, (2.8GHz, 50% Clk.) is not the lowest power state in our server. There are 3 lower power states, but a server hosting the TPC-W workload crashes if transitioned to any of these lower power states. These states are, therefore, considered infeasible for the TPC-W workload.

Power States (DVFS (GHz), Clk. Mod. (%))	(3.2, 100)	(2.8, 100)	(2.8, 50)
Normalized performance	1	1.18	15.69

Table 11: Performance degradation of TPC-W(60) at three different power states expressed as the ratio of average session response time with that offered by the servers operating at the highest power state (obtained from our offline profiling.) *Legend:* Perf.=Performance.

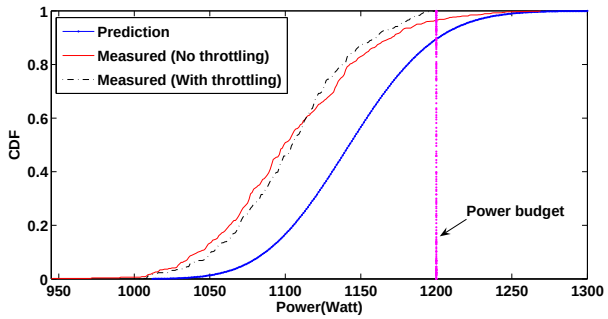


Figure 6: Sustained power profile (L=5 sec) for a 1200W PDU connected to 7 servers, each running TPC-W(60), with and without watermark-based enforcement.

revert back to their original power states.

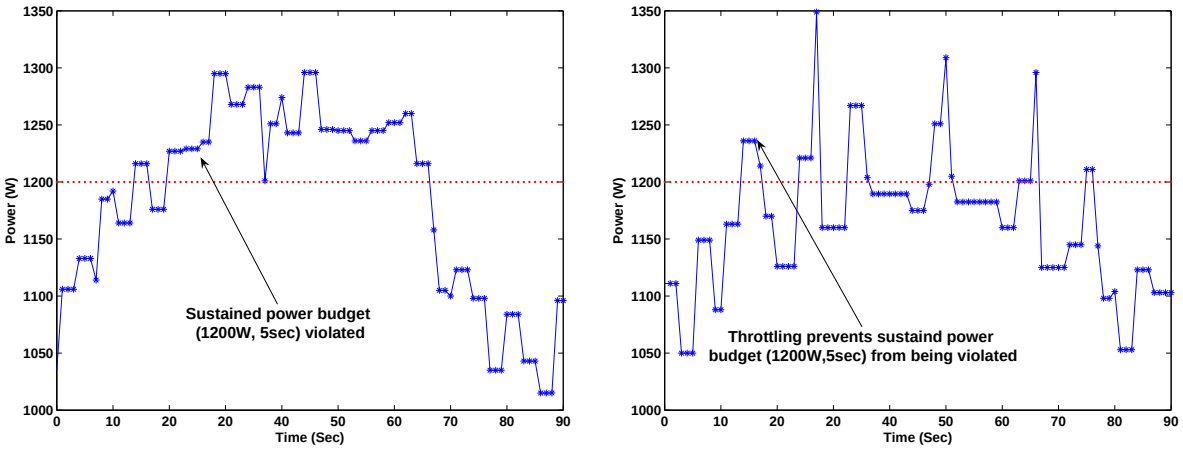
7.3 Performance Degradation

Next, we study any performance degradation resulting from the throttling necessitated by our aggressive provisioning. Using our prediction algorithm, we estimate the probability of the aggregate power consumption at the PDU exceeding its watermark. This probability, reported in Table 12, provides an estimate of the amount of time an application would find its server running at a throttled power state. We use our offline profiling technique (See Table 11) to estimate the performance degradation caused by different power states. We compare the predicted performance degradation with the measured values in Table 12. Since our watermark is (1200W, 3 sec), even if the application needs to consume 1200W or more all the time, it will be throttled only 60% of the time (3 seconds at the highest power state, 2 seconds at a throttled state). Therefore predicted watermark violation in Table 12 is computed by using the probability of violating the watermark (from predicted CDF) and then multiplying that probability by 0.6.

No. of Servers	Watermark Violation		Perf. Degradation		Feasible?
	Meas. (%)	Pred.(%)	Meas.	Pred.	
6	0	0	1	1	YES
7	2	7.2	1.04	1.08	YES
8	61.2	59.7	5.2	9.3	YES
9	N/A	N/A	N/A	N/A	NO

Table 12: Predicted and measured watermark violations at the PDU and normalized performance degradation for the instances of TPC-W(60), each running on one of the servers connected to the PDU. Performance degradation is expressed as the ratio of the average session response time with that offered by the server operating at its highest power state. The column labeled *Feasible?* indicates whether we would be able to prevent the sustained budget for the PDU from being violated. *Legend:* Meas.=Measured, Pred.=Predicted, and Perf.=Performance, N/A=Not Applicable.

In Table 12, our technique indicates that 7 is the most number of servers that could be safely connected to our PDU and still be operated to offer only a small performance degradation to overlying TPC-W applications. The predicted degradation was 1.88 while the measured degradation upon ac-



(a) Without throttling (b) With throttling

Figure 7: Power consumption recorded at the PDU connected to 7 servers, each running an instance of TPC-W(60), with and without the watermark-based budget enforcement.

tually connecting 7 servers running TPC-W(60) was only 1.04. We also estimate that while 8 servers can be safely connected and operated, such a configuration would result in significantly degraded performance (the measured normalized degradation, although much smaller than predicted, is still a significant 5.2.)

We take a closer look at application performance for the configuration connected 7 servers to the PDU that emerged as the most preferred in the discussion above. Figure 6 compares the sustained power consumption of a 1200W PDU consolidating 7 servers each running TPC-W(60), with and without throttling. Also shown is the predicted sustained power profile for this PDU. Figure 7 compares the power consumption of the PDU with and without the watermark-based throttling. We observe that our watermark enforcer is able to completely prevent sustained power violations by successfully identifying occasions at which the servers should be throttled.

7.4 Dynamic Changes in Workloads

We are interested in two aspects of dynamic variations exhibited either consume higher power than indicated by its power profile, which will result in more frequent violations of watermarks associated with power supply elements, raising the extent of degradation in performance (due to throttling.) Second, a workload may consume substantially lower power than its profile rendering provisioning more conservative. We evaluate the following simple mechanism to detect such changes and adjust the provisioning parameters accordingly.

We keep track of the recent power profile of the PDU and periodically compare it with its predicted profile. If there is a statistically significant difference between these distributions, (as reported by a well-regarded test such as the Kolmogorov-Smirnov Test [11]), we assume the workload has changed enough to necessitate re-provisioning. Note that upon detecting a phase change at the PDU level, we may want to percolate similar detection technique down the hierarchy to single out the application whose phase has changed. This can be achieved since we have the predicted profile of every server consolidated in the power hierarchy.

We evaluate a simple scenario to demonstrate the working of this mechanism. We consider a set

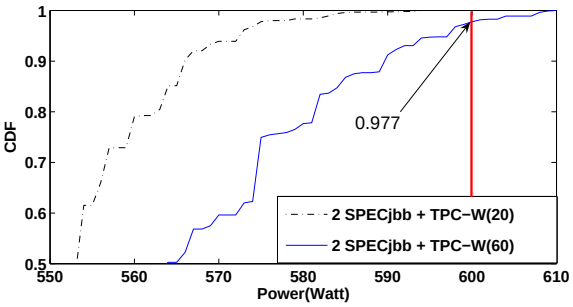


Figure 8: Illustration of the measured sustained power distributions of PDU_1 before and after the phase change. Workload change is detected by comparing the two distributions.

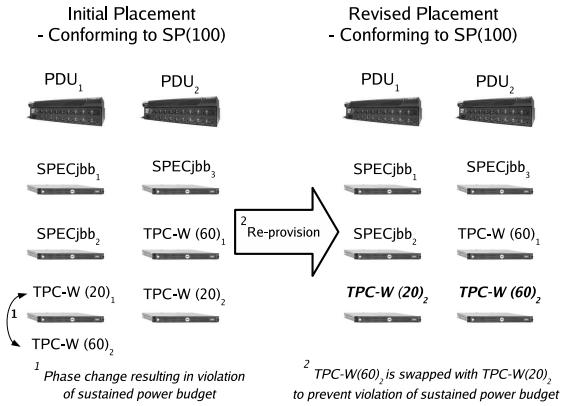


Figure 9: Illustration of phase change and consequent re-provisioning to prevent sustained power budget violation.

of six servers. Three of these servers run an instance of SPECjbb2005 each, two servers run TPC-W(20), and the sixth server runs TPC-W(60). We assume that we are provided with 2 PDUs (PDU_1 and PDU_2), each with a budget (600W, 5 sec). Based on our $SP(100)$ provisioning, we connect two SPECjbb2005 servers and a TPC-W(20) server to PDU_1. The remaining servers (hosting one out of SPECjbb2005, TPC-W(20), and TPC-W(60)) are connected to PDU_2. We assume that the workload of the TPC-W(20) connected to PDU_1 increases in intensity to TPC-W(60), simulating an overload where 40 new clients open sessions. The measured sustained power consumption of PDU_1, before and after this workload change, is presented in Figure 8. This triggers re-provisioning to accommodate the increased workload (the exact mechanisms of such re-provisioning are orthogonal to this work), that suggests the configuration change shown in Figure 9, where this instance of TPC-W(60) under PDU_1 is swapped with the TPC-W(20) under PDU_2 to prevent degraded performance for these new sessions via throttling.

We determine the overheads and effects of migration involved in the re-configuration described above. We use Xen's live migration facility [8] to migrate the TPC-W servers between the PDUs and find that it approximately takes 32 seconds to migrate the virtual machines. This causes a factor of 1.37 and 2.02 response time degradation for the workloads TPC-W(60) and TPC-W(20), respectively, during the transition.

8 Related Work

Research on Provisioning of Power Infrastructure. *Server-level:* Server vendors (including IBM, HP, Sun, and Dell [13]), in an attempt to help data centers administrators do better provisioning, provide calculators for estimating the peak power needs of their servers. Such calculators provide worst-case power estimates for a server given its component configuration and workload characteristics. Lefurgy et al. [24] observe that the power supplies of servers are typically over-provisioned and report that replacing these with cheaper power supplies with 15% lower capacities results in negligible degradation in performance. To prevent rare power spikes from exceeding the capacity, they implement a reactive technique similar to ours that keeps the server power within safe limits. Felter et al. [16] observe that different components of a server (CPU, memory, etc.) do not require their peak power simultaneously. That is, the statistical multiplexing effects explored in our paper extend even to the granularity of disparate resources within a server. They devise a technique that dynamically proportions the total system power among the system components thereby reducing the power and cooling requirements of a server.
Cluster-level: Ensemble-level power management [28] by Ranganathan et al. looks at provisioning the cooling capacity as opposed to our work that looks at provisioning power capacity expended towards operating servers. The authors observe that for real workloads, the possibility of their power consumption happening simultaneously is small and use it to better provision the cooling capacity at a blade enclosure level. Their results showed that they are able to realize a reduction of 20% system power budget with negligible performance impact. Very closely related work of our research is recent work by Fan et al. [15] which also looks at provisioning the power infrastructure for large data centers at different levels of the power hierarchy. They analyze data from a real large-scale Google data center and observe that 40% additional servers can be accommodated within the power supply of their data center. While these observations motivate our research, we take them one step further and devise a methodical approach to characterize and statistically understand applications' power usage and then use it for provisioning the power infrastructure.

While the basic idea of the above techniques (both at the server level as well ais similar to our contention (exploiting the gap between provisioned capacity and actual peak power needs of applications), in the best While our research shares several basic ideas with the above body of work, to the best of our knowledge, we are the first ones to: (i) employ the notion of sustained power budgets at PDUs and beyond to capture the safety concerns for their fuses/circuit breakers, (ii) characterize sustained power needs of individual applications and use them to estimate sustained power behavior of aggregates, and (iii) use devise provisioning techniques that can employ these estimates to strike the desired balance between cost gains and performance loss while guaranteeing safe operation.

Control Techniques for Power/Performance Trade-offs. CPU throttling has widely been adopted for enforcing peak power budgets related to: (a) thermal constraints/cooling capacity of a server [27] and (b) provisioning power capacity of a server [24, 15]. Recent work by Wang et al. [42] develops a control-theoretic model that enforces a specified power budget at a cluster level and dynamically distributes the power budget among the connected servers based on their needs. New IBM servers [21] are shipped with capability to enforce a specified power budget at very fine time granularity (in order of msec) by using cpu throttling. Nathuji et al. [25] extend power management solutions for the virtual machines running on virtualized hardware. Raghavendra et al. [27] look at co-ordinating the different power budgets (peak and average power budget) enforced at different granularities. Lot of research

has gone into evaluating the energy/performance trade-offs of applications which involves keeping either energy or performance as a constant and optimizing for the other metric [43, 5]. The above techniques for implementing some form of power budgets are complementary to our work.

Yield Management in Other Areas. Yield management (YM) was first explored in the airline industry, particularly by American Airlines [32]. YM-inspired practices have subsequently been also explored in areas such as telephony and networking [18, 40, 4], memory management [41], and CPU/network management [39] for servers. While the general principles of under-provisioning, statistical multiplexing, and overbooking are well-explored in these diverse contexts, their application to power infrastructure presents some unique challenges. Such techniques must be necessarily accompanied by effective protective mechanisms such as those developed in Section 5.1 since the consequences of violation are disastrous. Also, the hierarchical nature of the power infrastructure implies that the interactions between decisions taken at various levels must be taken into account. While we restricted our efforts to investigating the impact of under-provisioning at a single level (a PDU supplying power to multiple servers), developing a *comprehensive understanding of provisioning parameters for the entire power hierarchy* is a direction for future research.

9 Concluding Remarks

The central thesis of this research was that by carefully understanding the power needs of hosted workloads and their aggregates, a data center could significantly improve the cost-revenue trade-off associated with its power supply hierarchy—ranging from server-level power supplies to higher-level Power Distribution Units (PDUs) and Uninterrupted Power Supply sub-stations. Towards this end, we developed a measurement technique to derive power profiles of applications. Our profiles succinctly captured key statistical properties of the power usage of applications and lent themselves to the design of power usage predictors for aggregates of these applications. We designed a novel technique guided by these power profiles that employed controlled under-provisioning, statistical multiplexing, and overbooking when provisioning the power infrastructure. Our evaluation on a prototype data center using well-regarded benchmarks demonstrated the feasibility and benefits of our technique. As a representative result, by accurately identifying the worst-case needs of hosted workloads, our technique was able to improve the number of servers that could be safely connected to a PDU by 150% compared to the currently prevalent practice of using face-plate ratings. Exploiting statistical multiplexing among the power usage of these servers along with controlled under-provisioning based on tails of power profiles offered a further gain of 100% over face-plate provisioning. Furthermore, evidence of self-similarity in the power usage of some workloads suggests that such gains can be expected even higher up in the power hierarchy. Reactive techniques implemented in the Xen VMM running on our servers dynamically modulated CPU DVFS-states to contain power draw within safe limits despite our aggressive provisioning. Finally, information yielded by our profiles also provided ways of controlling the performance degradation resulting from our under-provisioning: e.g., the 95^{th} percentile of response time of TPC-W sessions grew from 1.59 sec to 1.78 sec, a degradation of only 11.65%.

10 Availability

A detailed technical report describing the sustained power prediction technique employed in this research [7], a Xen patch for enabling MSR writes and implementing our watermark-based budget enforcement mechanism, and all experimental data are available at: `http://csl.cse.psu.edu/hotmap`.

References

[1] J. Anderson, L. Berc, J. Dean, S. Ghemawat, M. Henzinger, S. Lueng, M. Vandervoorde, C. Waldspurger, and W. Weihl. Continuous Profiling: Where Have All the Cycles Gone? In *Proceedings of the 16th ACM Symposium on Operating Systems Principles*, pages 1–14, October 1997.

[2] P. Barham, B. Dragovic, K. Fraser, S. Hand, T. Harris, A. Ho, R. Neugebauer, I. Pratt, and A. Warfield. Xen and the art of virtualization. In *SOSP '03: Proceedings of the nineteenth ACM symposium on Operating systems principles*, pages 164–177, New York, NY, USA, 2003. ACM.

[3] M. Benani and D. Menasce. Resource Allocation for Autonomic Data Centers Using Analytic Performance Models. In *Proceedings of IEEE International Conference on Autonomic Computing, Seattle (ICAC-05), WA* , June 2005.

[4] R. Boorstyn, A. Burchard, J. Liebeherr, and C.Oottamakorn. Statistical Service Assurances for Traffic Scheduling Algorithms. In *IEEE Journal on Selected Areas in Communications, 18:12*, pages 2651–2664, December 2000.

[5] S. Chaitanya, B. Urgaonkar, and A. Sivasubramaniam. QDSL: QoS-aware Systems with Differential Service Levels. In *Proceedings of the ACM Conference on Measurement and Modeling of Computer Systems (SIGMETRICS 2008), Annapolis, MD, to appear*, June 2008.

[6] Y. Chen, A. Das, W. Qin, A. Sivasubramaniam, Q. Wang, and N. Gautam. Managing Server Energy and Operational Costs in Hosting Centers. In *Proceedings of the ACM International Conference on Measurement and Modeling of Computer Systems (SIGMETRICS 2005), Banff, Canada, June 2005*, June 2005.

[7] J. Choi, S. Govindan, B. Urgaonkar, and A. Sivasubramaniam. Profiling, Prediction, and Capping of Power Consumption in Consolidated Environments. Technical Report CSE08-006, The Pennsylvania State University, Apr. 2008. `http://www.cse.psu.edu/~sgovinda/PowerPredictionTR.pdf`.

[8] C. Clark, K. Fraser, S. Hand, J. Hansen, E. Jul, C. Limpach, I. Pratt, and A. Warfield. Live Migration of Virtual Machines. In *Proceedings of the Second Symposium on Networked Systems Design and Implementation (NSDI'05)*, May 2005.

[9] D. Clark. Power Hungry Computers Put Data Centers in Bind. *The Wall Street Journal (Online)*, November 2005. `http://hightech.lbl.gov/DCTraining/docs/wsjon-data-ctr-power.pdf`.

[10] I. Cohen, J. Chase, M. Goldszmidt, T. Kelly, and J. Symons. Correlating Instrumentation Data to System States: A Building Block for Automated Diagnosis and Control. In *Proceedings of the Sixth USENIX Symposium in Operating Systems Design and Implementation (OSDI 2004), San Francisco, CA*, December 2004.

[11] P. Cohen. *Empirical Methods for Artificial Intelligence*. MIT Press, Cambridge, MA, USA, 1995.

[12] Dell Computers: PowerEdge servers SC1425 Spec Sheet, Dec 2005. `http://www.dell.com/downloads/global/products/pedge/en/sc1425_specs.pdf%`.

[13] Dell Server Power Calculator, May 2008. `http://www.dell.com/content/topics/topic.aspx/global/products/pedge/top%ics/en/config_calculator?c=us\&cs=555\&l=en\&s=biz`.

[14] R. Doyle, J. Chase, O. Asad, W. Jin, and A. Vahdat. Model-Based Resource Provisioning in a Web Service Utility. In *Proceedings of the Fourth USITS*, Mar. 2003.

[15] X. Fan, W.-D. Weber, and L. A. Barroso. Power Provisioning for a Warehouse-Sized Computer. In *Proceedings of the Thirty Fourth Annual International Symposium on Computer Architecture*, 2007.

[16] W. Felter, K. Rajamani, C. Rusu, and T. Keller. A Performance-Conserving Approach for Reducing Peak Power Consumption in Server Systems. In *ICS 2005*, June 2005.

[17] Gartner, 2007 Press Release. `http://www.globalactionplan.org.uk/upload/resource/Full-report.pdf`.

[18] A. Gupta, D. Stahl, and A. Whinston. The Economics of Network Management. *Communications of the ACM*, 42(5):57–63, 1999.

[19] S. Heins. Divorcing Electricity Sales from Profits Creates Win-Win for Utilities and Customers, Energy Pulse, Sept. 2006. `http://www.energypulse.net/centers/article/article_display.cfm?a_id=134%2`.

[20] J. Hellerstein, F. Zhang, and P. Shahabuddin. A Statistical Approach to Predictive Detection. *Computer Networks*, Jan. 2000.

[21] IBM Active Energy Manager - Measure and Cap Energy. `http://www-03.ibm.com/press/us/en/pressrelease/22551.wss`.

[22] R. Iyer, V. Tewari, and K. Kant. Overload Control Mechanisms for Web Servers. In *Workshop on Performance and QoS of Next Generation Networks*, Nov. 2000.

[23] M. Lavelle. Conservation Can Mean Profits for Utilities: States are changing the rules of the game so that it pays power companies not to expand, USNews and World Report, May 2008. `http://www.usnews.com/`.

[24] C. Lefurgy, X. Wang, and M. Ware. Server-Level Power Control. In *ICAC '07: Proceedings of the Fourth International Conference on Autonomic Computing*, page 4, Washington, DC, USA, 2007. IEEE Computer Society.

[25] R. Nathuji and K. Schwan. Virtualpower: Coordinated power management in virtualized enterprise systems. In *21st ACM Symposium on Operating Systems Principles (SOSP'07)*, 2007.

[26] K. Park and W. Willinger, editors. *Self-Similar Network Traffic and Performance Evaluation*. Wiley-Interscience, John Wiley and Sons, Inc., 2000.

[27] R. Raghavendra, P. Ranganathan, V. Talwar, Z. Wang, and X. Zhu. No Power Struggles: Coordinated multi-level power management for the data center. In *Thirteenth International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS '08)*, Mar. 2008.

[28] P. Ranganathan, P. Leech, D. Irwin, and J. Chase. Ensemble-level Power Management for Dense Blade Servers. In *Proceedings of the International Symposium on Computer Architecture (ISCA)*, June 2006.

[29] Raritan 20Amp PDU Model - DPCR 20-20, May 2008. `http://www.raritan.com/products/power-management/Dominion-PX/DPCR20-20/%`.

[30] S. Shende, A. Malony, J. Cuny, K. Lindlan, P. Beckman, and S. Karmesin. Portable Profiling and Tracing for Parallel Scientific Applications using C++. In *Proceedings of ACM SIGMETRICS Symposium on Parallel and Distributed Tools (SPDT)*, pages 134–145, August 1998.

[31] Signametrics Multimeter SM2040 Series Spec Sheet, May 2000. `http://www.signametrics.com/products/sm2040.pdf`.

[32] B. C. Smith, J. F. Leimkuhler, and R. M. Darrow. Yield Management at American Airlines. In *Interfaces, 22:1*, pages 8–31, January-February 1992.

[33] W. Smith. TPC-W: Benchmarking An Ecommerce Solution. `http://www.tpc.org/information/other/techarticles.asp`.

[34] The Standard Performance Evaluation Corporation (SPEC). `http://www.spec.org/`.

[35] SPEC JBB2005: Java Business Benchmark. `http://www.spec.org/jbb2005/`.

[36] C. Stewart, T. Kelly, and A. Zhang. Exploiting Nonstationarity for Performance Prediction. In *Proceedings of EuroSys2007, Lisbon, Portugal*, March 2007.

[37] C. Stewart and K. Shen. Performance Modeling and System Management for Multi-component Online Services. In *Proceedings of the Second USENIX Symposium on Networked Systems Design and Implementation (NSDI'05), Boston MA*, pages 71–84, May 2005.

[38] B. Urgaonkar, G. Pacifici, P. Shenoy, M. Spreitzer, and A. Tantawi. An Analytical Model for Multi-tier Internet Services and its Applications. In *Proceedings of the ACM International Conference on Measurement and Modeling of Computer Systems (SIGMETRICS 2005), Banff, Canada*, June 2005.

[39] B. Urgaonkar, P. Shenoy, and T. Roscoe. Resource Overbooking and Application Profiling in Shared Hosting Platforms. In *Proceedings of the Fifth USENIX Symposium on Operating Systems Design and Implementation (OSDI 2002), Boston, MA*, December 2002.

[40] H. M. Vin, P. Goyal, A. Goyal, and A. Goyal. A Statistical Admission Control Algorithm for Multimedia Servers. In *Proceedings of the ACM Multimedia'94, San Francisco, CA*, pages 33–40, October 1994.

[41] C. Waldspurger. Memory Resource Management in VMWare ESX Server. In *Proceedings of the Fifth Symposium on Operating System Design and Implementation (OSDI'02)*, December 2002.

[42] X. Wang and M. Chen. Cluster-level feedback power control for performance optimization. In *Proceedings of the Fourteenth International Symposium on High-Performance Computer Architecture (HPCA'08)*, Feb. 2008.

[43] A. Weisel and F. Bellosa. Process cruise control-event-driven clock scaling for dynamic power management. In *Proceedings of the International Conference on Compilers, Architecture and Synthesis for Embedded Systems (CASES 2002)*, Oct. 2002.

[44] H. Zeng, X. Fan, C. Ellis, A. Lebeck, and A. Vahdat. ECOSystem: Managing Energy as a First Class Operating System Resource. In *Proceedings of the Architectural Support for Programming Languages and Operating Systems(ASPLOS)*, October 2002.