Towards Energy-Proportional Computing for Enterprise-Class Server Workloads

Balaji Subramaniam and Wu-chun Feng
Dept. of Computer Science
Virginia Tech
{balaji, feng}@cs.vt.edu

ABSTRACT

Massive data centers housing thousands of computing nodes have become commonplace in enterprise computing, and the power consumption of such data centers is growing at an unprecedented rate. Adding to the problem is the inability of the servers to exhibit *energy proportionality*, i.e., provide energy-efficient execution under all levels of utilization, which diminishes the overall energy efficiency of the data center. It is imperative that we realize effective strategies to control the power consumption of the server and improve the energy efficiency of data centers. With the advent of Intel Sandy Bridge processors, we have the ability to specify a limit on power consumption during runtime, which creates opportunities to design new power-management techniques for enterprise workloads and make the systems that they run on more *energy proportional*.

In this paper, we investigate whether it is possible to achieve *energy proportionality* for an enterprise-class server workload, namely SPECpower_ssj2008 benchmark, by using Intel's Running Average Power Limit (RAPL) interfaces. First, we analyze the power consumption and characterize the instantaneous power profile of the SPECpower benchmark within different subsystems using the *on-chip energy meters* exposed via the RAPL interfaces. We then analyze the impact of RAPL *power limiting* on the performance, per-transaction response time, power consumption, and energy efficiency of the benchmark under different load levels. Our observations and results shed light on the efficacy of the RAPL interfaces and provide guidance for designing power-management techniques for enterprise-class workloads.

Categories and Subject Descriptors

C.0 [**Computer Systems Organization**]: General—*System architectures*; C.4 [**Computer Systems Organization**]: Performance of systems—*Design studies, Measurement techniques*

ICPE'13, April 21–24, 2013, Prague, Czech Republic.

General Terms

Measurement, Experimentation

Keywords

Power Limiting, Energy Proportionality, Greenness

1. INTRODUCTION

Massive data centers, which house thousands of computing nodes, have become increasingly more common. A large fraction of such data centers' total cost of ownership (TCO) comes from the cost of building and maintaining infrastructure that is capable of powering such large-scale data centers and from the recurring energy costs [13]. Consequently, power and energy have emerged as first-order design constraints in data centers. These issues are further magnified by the inability of servers to provide energy-efficient execution at all levels of utilization. The recent recommendation of *energy proportionality* in servers, i.e., to design servers that consume power proportional to the utilization, is a move in the right direction as it has the potential to double the energy efficiency of servers [10]. However, achieving energy-proportional operation is a challenging task, particularly given that typical servers consume 35-45% of peak power even when idling.

Typically, dynamic voltage and frequency scaling (DVFS) has been used to achieve better energy efficiency as it can potentially give up to cubic energy savings [18, 25, 27]. However, as we will show in this paper, the subsystem affected by DVFS (i.e., the *core*[1]) is already the most energy-proportional part of the system. There are other subsystems, such as the *uncore*,[2] that consume constant power, irrespective of the system utilization. In order to achieve energy proportionality, we need to understand the power consumption of each subsystem at different levels of utilization and to leverage mechanisms that enable us to control the power consumption of these subsystems.

With the advent of Intel Sandy Bridge processors, we have better control over the power consumption of the system via the running average power limit (RAPL) interfaces [12, 1]. RAPL exposes *on-chip energy meters* for the *core* subsystem, processor package, and DRAM and enables the tracking of power consumption at a time resolution (∼1 ms) and

[1]The *core* subsystem includes components such as the ALUs, FPUs, L1, and L2 caches [2].
[2]The *uncore* subsystem includes components such as the memory controller, integrated I/O, and coherence engine [2].

system-level granularity that was not possible before. Moreover, it facilitates deterministic control over the power consumption of subsystems through *power-limiting* interfaces. These interfaces allow a user to specify a power bound and a time window over which the bound should be maintained. While this hardware-enforced power limiting is an appealing option, the impact of power limiting on the performance, power, and energy efficiency of enterprise-class server workloads is still not well understood and remains an active area of research.

In this paper, we investigate whether it is possible to achieve *energy-proportional* operation for an enterprise-class server workload, namely the SPECpower_ssj2008 benchmark (henceforth referred to as SPECpower) by using the RAPL interfaces. To this end, this paper makes the following contributions:

- Insights into the mechanisms of power management for enterprise-class server workloads using the RAPL interfaces via an analysis of the SPECpower benchmark by calibrating its input parameters.
- A rigorous quantification of the energy proportionality of each subsystem within a server node via an analysis of power-consumption profiles of the different subsystems when running SPECpower at different load levels.
- The identification of system activity based on performance counter traces that strongly correlates with subsystem-level power consumption captured via the RAPL interfaces.
- An analysis and characterization of the instantaneous-power profiles at different load levels of SPECpower to understand the time resolution at which power limiting should be applied.
- Empirical results on the impact of RAPL power limiting on power, performance, per-transaction response time, and energy efficiency.

Through our contributions, we make the following observations and conclusions on the power management of the SPECpower benchmark using RAPL interfaces. First, the *core* is the most energy-proportional subsystem and the *uncore* is the least. Second, there is the opportunity for limiting the power consumption of the *core* and processor package subsystems at a resolution less than 50 ms for different configurations of SPECpower. Third, as the load level increases, the ability to limit the power consumption at the DRAM-level decreases. Fourth, power limiting at the level of the *core* subsystem is the best option for improving energy efficiency and achieving energy proportionality. Fifth, even when power limiting at the processor package-level, most of the power savings comes from the *core* subsystem. Sixth, better power-management mechanisms are required to achieve energy proportionality at the *uncore* subsystem-level. Seventh, though we were not able to achieve energy proportionality at the full system level, i.e., entire compute node, we show that energy-proportional operation is possible at the granularity of subsystems over which we have control via RAPL power limiting (i.e., *core* subsystem, processor package, and DRAM).

The rest of the paper is organized as follows. In Section 2, we present the details of the SPECpower benchmark and Intel RAPL interfaces. Section 3 describes our analysis and characterization of average and instantaneous power consumption of all subsystems at different load levels in SPECpower and the observations from these experiments. Next, in Section 4, we limit the power consumption of SPECpower within different subsystems and present the impact of power limiting on power, performance, and energy efficiency. In Section 5, we describe the related work, and we conclude in Section 6.

2. BACKGROUND

In this section, we provide an overview of the SPECpower benchmark and its design as well as details into its configurable parameters. We then present the control and capabilities exposed by Intel's RAPL interfaces.

2.1 Overview of SPECpower Benchmark

SPECpower [4] is an industry-standard benchmark that measures both the power and performance of a server node. The benchmark mimics a server-side Java transaction processing application and is based on the SPECjbb2005 benchmark [3]. It stresses the CPU, caches, and memory hierarchy and tests the implementations of the Java virtual machine (JVM), just-in-time (JIT) compiler, garbage collection, and threads. The benchmark requires two systems: (1) the system under test or SUT and (2) the control and collection system (CCS) [6] with communication between the systems established via Ethernet [7].[3] The SUT runs the workload and is connected to a power meter. The power meter, in turn, is connected to the CCS. The CCS collects the performance and power data passed to it by the SUT and power meter, respectively.

The SPECpower benchmark is designed to produce consistent and repeatable performance and power measurements. It executes different type of transactions and the transactions are grouped together in *batches* for scheduling purposes. Each *load-level* is achieved by controlling delay between the arrival of batches.

More specifically, the SPECpower benchmark is a graduated workload, i.e., it runs the workload at different *load-levels* and reports the power and performance at each load-level [9]. The benchmark starts with a calibration phase, which determines the maximum throughput. The calibrated throughput is set as the throughput target for 100% load-level. The throughput target for the rest of the load-levels is calculated as a percentage of the throughput target for 100% load-level. For example, if the throughput target for 100% load-level is 100,000, then the target for 70% load-level is 70,000, 40% is 40,000 and so on. The throughput is measured in server-side Java operations per second (ssj_ops).

The benchmark supports a set of configurable parameters.[4] For example, the maximum target throughput and the batch size can be manually configured. We refer the reader to [8] for further information on configurable parameters. The flexibility, coupled with the consistency and repeatability of SPECpower, allows us to evaluate the applicability of newer power-management interfaces, such as RAPL, to enterprise-class server workloads.

[3]SUT and CCS can be the same system. Communication is established via Ethernet only if the systems are different.

[4]Only a subset of these parameters can be changed for compliant runs.

2.2 Intel's Running Average Power Limit (RAPL) Interfaces

RAPL was introduced in Intel Sandy Bridge processors. The RAPL interfaces provide mechanisms to enforce power consumption limits on a specific subsystem. The only official documentation available for these interfaces is section 14.7 of the Intel software developer's manual [1]. Our experiments deal only with the Sandy Bridge server platforms.

The RAPL interfaces can be programmed using the model-specific registers (MSRs). MSRs are used for performance monitoring and controlling hardware functions. These registers can be accessed using two instructions: (1) *rdmsr*, short for "read model-specific registers" and (2) *wrmsr*, short for "write model-specific registers." The *msr* kernel module can be used for accessing MSRs from *user space* in Linux environments. When loaded, the *msr* module exposes a file interface at */dev/cpu/x/msr*. This file interface can be used to read from or write to any MSR on that CPU.

According to the Intel documentation, RAPL interfaces operate at the granularity of a processor socket. The server platforms provide control over three domains (i.e., subsystems):[5] (1) package (PKG), (2) power plane 0 (PP0), and (3) DRAM. We expect PKG, PP0 and DRAM to represent the processor package (or socket), the *core* subsystem, and memory DIMMs associated with that socket, respectively. The *MSR_RAPL_POWER_UNIT* register contains the units for specifying time, power, and energy, and the values are architecture-specific. For example, our testbed requires and reports time, power, and energy at increments of 976 microseconds, 0.125 watts, and 15.3 microjoules, respectively. Each domain consists of its own set of RAPL MSR interfaces. On a server platform, RAPL exposes four capabilities:

1. Power limiting – Interface to enforce limits on power consumption.
2. Energy metering – Interface reporting actual energy usage information.
3. Performance status – Interface reporting performance impact due to power limit.
4. Power information – Interface which provides value range for control attributes associated with power limiting.

2.2.1 Power Limiting

RAPL maintains an average power limit over a sliding window instead of enforcing strict limits on the instantaneous power. The advantage of having an average power limit is that if the average performance requirement is within the specified power limits the workload will not incur any performance degradation even if the performance requirement well exceeds the power limit over short bursts of time. The user has to provide a power bound and a time window in which the limit has to be maintained. Each RAPL domain exposes a MSR which is used for programming these values. The PKG domain provides two power limits and associated time window for finer control over the workload performance whereas other domains provide only one power limit. The interface provides a *clamping* ability, which when enabled, allows the processor to go below an OS-requested P-state.

[5]Note: We use RAPL domain and subsystem interchangeably in rest of the paper.

Table 1: Per-Socket Parameter Range (MTW = Maximum Time Window, MaxP = Maximum Power, MinP = Minimum Power). Multiply by 2 for Full Two-Socket System.

Domain/Range	MTW	MaxP	MinP
Package	45.89 ms	180 watts	51 watts
DRAM	39.06 ms	75 watts	15 watts

2.2.2 Energy Metering

Each domain exposes a MSR interface that reports the energy consumed by that domain. On a server platform, we expect the that (1) *energy(PKG) = energy consumed by the processor package*, (2) *energy(PP0) = energy consumed by the core subsystem*, and (3) *energy(uncore subsystem) = energy(PKG) − energy(PP0).*

2.2.3 Performance Status

This MSR interface reports the total time for which each domain was throttled (i.e., functioning below the OS-requested P-state) due to the enforced power limit. This information will be useful in understanding the effects of power limiting on a particular workload.

2.2.4 Power Information

The PKG and DRAM domains expose a MSR interface that provides information on the ranges of values that can be specified for a particular RAPL domain for limiting its power consumption. This includes maximum time window, maximum power, and minimum power. The range of per-socket values on our experimental platform is given in Table 1.

3. AN ANALYSIS OF POWER CONSUMPTION FOR SPECPOWER

In this section, we characterize the power consumption of the SPECpower benchmark and analyze energy proportionality from the perspective of the entire system as well as each RAPL domain. We then analyze performance counter traces to understand the trends seen in power consumption. Finally, we present our characterization of the instantaneous power profile of SPECpower and provide insights into the time window used for limiting the power consumption of the benchmark under different configurations.

We will also show the following: (1) the most and least energy-proportional subsystems are the *core* (PP0) and the *uncore* (Package-PP0), respectively, (2) there is ample opportunity to limit the power consumption of SPECpower at different load-levels below the 50-ms resolution for the package and PP0 domains, and (3) there is little opportunity to limit the power consumption at the DRAM-level as the load-level increases.

3.1 Experimental Setup

The SUT for our experiments is an Intel Xeon E5-2665 processor (Intel Romley-EP). The node has two such processors for a total of 16 cores and 32 cores when hyper-threading is ON, as in our experiments. It has 256 GB of

Table 2: Performance Counters Used

Last-level cache references/second (LLCR)	Last-level cache misses/second (LLCM)
Instructions retire/second (IR)	Unhalted core clock cycles (UC)
Number of offcore outstanding reads/second (NOUT)	Cycles in which there was one or offcore outstanding reads (COUT)
Misses in DTLB loads that cause a page walk/second (DTLBLM)	Misses in DTLB stores that cause a page walk/second (DTLBSM)
Cycles spent in page walks due to DTLB load misses (DTLBLC)	Cycles spent in page walks due to DTLB store misses (DTLBSC)
Misses in ITLB that cause a page walk/second (ITLBM)	Cycles spent in page walks due to ITLB misses (ITLBC)

memory and runs a Linux kernel version 3.2.0. The CCS has an Intel Xeon E5405 processor with 8 cores and 8 GB of RAM. The CCS runs a Linux kernel version 2.6.32. The CCS and SUT were connected through a gigabit Ethernet network. To measure power, we used a Yokogawa WT210 power meter.

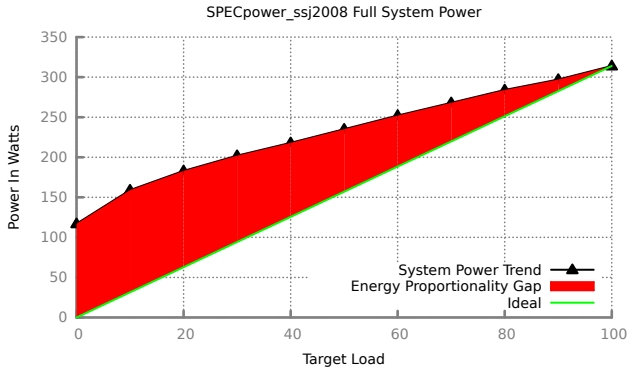

Figure 1: Illustration of Energy Proportionality Gap

We used all the cores in SUT for our experiments. Eight warehouses with four threads for each warehouse were used as the configuration for SPECpower. The four threads in each warehouse were pinned to two adjacent physical cores on the SUT using *numactl*. In order to provide consistent performance results throughout our experiments, we configured the *input.load_level.target_max_throughput* parameter to achieve the same performance for each run. It was set to 115,375 ssj_ops for each warehouse for a total of 923,000 ssj_ops for the entire run. In all our experiments, 100% load-level corresponds to 923,000 ssj_ops. This value was determined by averaging 10 calibration runs.

We changed the runtime for each load-level to 120 seconds using the *input.load_level.length_seconds* parameter and the pre- and post-measurement interval to 15 seconds in order to reduce the total runtime of the benchmark. On average, our testbed consumes 117 watts at idle and 314 watts at 100% load-level of SPECpower. We would like to stress that the system consumes 37.2% of peak power when idling.

3.2 Energy Proportionality Metric

As discussed earlier, we are interested in analyzing the energy proportionality of the system. The deviation of the power curve of the system from the ideal power curve is of particular interest to us. To illustrate with an example, Figure 1 shows the average power consumed by the testbed at each workload and the hypothetical/ideal energy proportional power curve for the system. The red shaded region is the deviation of interest. Henceforth, this area will be referred to as *energy proportionality gap (EPG)*. We quantify the EPG using the EP metric. The EP metric is calculated as one minus the ratio of EPG (the red region in the figure) and the area under the ideal curve [26]. A value of 1 for the metric represents an ideal energy-proportional system. A value of 0 represents a system that consumes a constant amount of power irrespective of the percentage of load-level. In our discussions, we determine the metric by calculating the area under curves. In particular, we calculate the area under the curve as the sum of areas of rectangles.

3.3 Analysis of Subsystem-Level Power Consumption

In this section, we present the details on the power consumption of SPECpower at a subsystem-level. We were able to profile the benchmark at a granularity that has not been possible until the advent of Intel Sandy Bridge by using the on-chip energy meters exposed by the RAPL interfaces. Our results provide insights into the energy proportionality of a system as a whole as well as at the RAPL domain-level. We also present the details of the power consumed using different batching sizes. Batching queries to exploit and create opportunities for power management is a well-researched area [21]. The number of transactions in each batch scheduled of the SPECpower benchmark is calibrated using the *input.scheduler.batch_size* input parameter. We use four different batching sizes (i.e, 1,000, 5,000, 7,500 and 10,000) in each of our experiments.

3.3.1 Methodology

We used the energy meters exposed in each RAPL domain to determine the power dissipated in each domain. In all our results, we report the average power[6] over five runs for the domain-level power consumption. For full-system power measurement, we have followed the power measurement methodology specified and developed by the SPEC organization for the SPECpower benchmark [5]. We also collected specific performance counter data that correspond to RAPL domain power and full system power in order to deliver more insights into the power profile of SPECpower. Table 2 shows the list of performance counters we profiled. We determined these performance counters based on previous work [23, 15], and we use newer ones that might have effects on the power curve of the system.

Because we have access to energy meters at different RAPL domains, we are also interested in finding the performance

[6]Average power is calculated as (initial energy reading - final energy reading)/time.

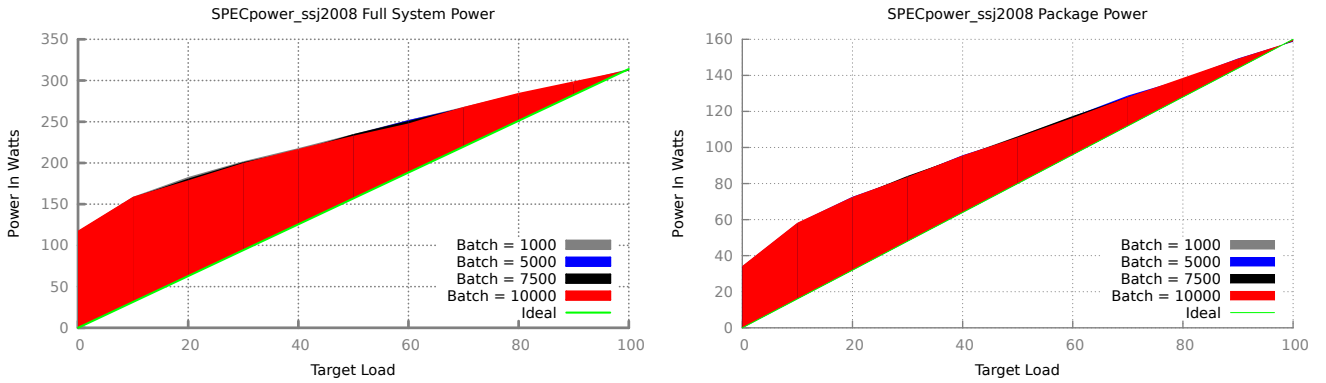

Figure 2: Power Profiles – Left:Full System, Right:Package. Both power trends are similar indicating a strong correlation between full system and package power consumption.

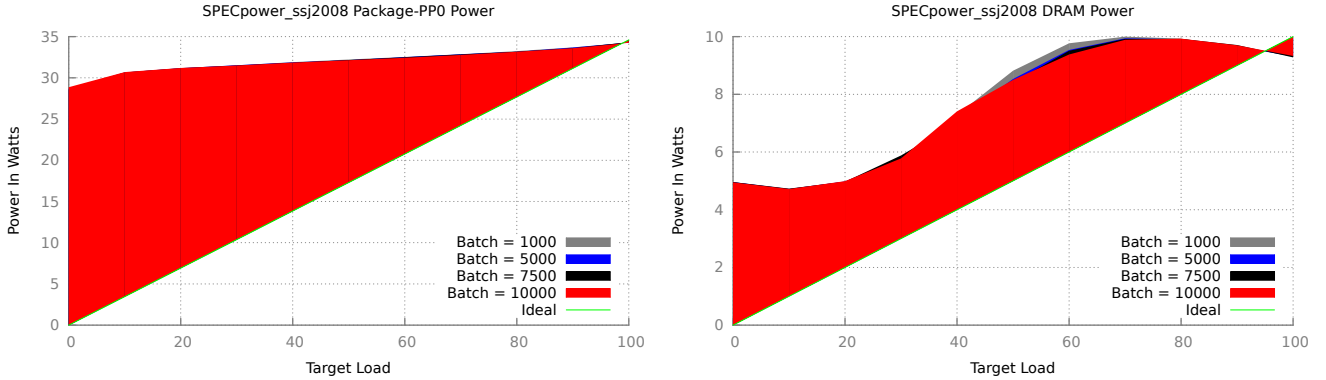

Figure 3: Power Profiles – Left: *Uncore* (Package-PP0), Right:DRAM. *Uncore* power is constant irrespective of the load-level making it the least energy-proportional subsystem.

counters that affect the power curve of that domain. Specifically, we want to find the performance counters that correlate to the power consumption of each domain. This study will help guide the modeling of the power consumption of different load-levels at a subsystem-level. Principal component analysis (PCA) is used to determine the most significant performance counter for a particular RAPL domain as well as the full system. We perform PCA on system activity to provide a concise and statistically meaningful overview of the causes for the trends in the subsystem as well as the full system power trend. Note that the correlation analysis is performed before performing PCA in order to obtain performance counters that show good correlation to power.

PCA is widely used for dimension reduction in many areas. The main concept behind PCA is to reduce the number of dimensions in a data set while preserving the variance of the data set as much as possible. This is achieved by transforming the data into principal components (PCs). The principal components are linear combinations of the original set of variables and are uncorrelated to each other. We first normalize the performance counter data to a unit normal distribution to mitigate the effect of the varying range of values for different performance counters and then apply PCA. We use R to perform these statistical analyses in our experiments.

3.3.2 *Profiling of Full System and Package Power*

Figure 2 shows the EPGs for the full system and package RAPL domain while running SPECpower using different batch sizes. In both cases, the batch size does not affect the average power profile of the system. It is also worth noting that both the power profile follow a similar trend, indicating a strong correlation between full system and package power. (The Pearson correlation is greater than 0.99.) Consequently, the power consumed by the domains have a strong correlation (> 0.95) with the same performance counters. Both these power profiles have a strong correlation with all counters except the counters related to ITLB in Table 2. Our PCA reveals that five PCs (LLCM, IR, NOUT, DTLBLM and DTLBSM) account for 95% of the variance.

The EP metric value for the full system is 0.54 for all batch sizes whereas the metric value for the package domain is 0.71 for all batch sizes. Comparison of the EP metric shows that EPG for the package domain is smaller than the full system (recall that larger is better for EP and smaller is better for EPG) suggesting that the package domain is more energy-proportional than the full system.

3.3.3 *Profiling of Uncore (Package-PP0) and DRAM Power*

Figure 3 shows the *uncore* subsystem and DRAM EPGs.

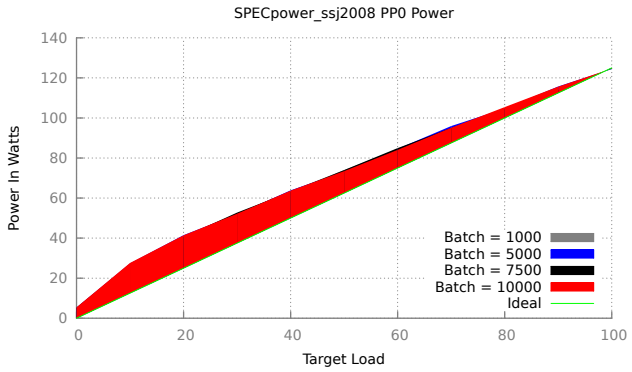

Figure 4: Power Plane 0 (PP0) Power Profile. PP0 subsystem has the smallest EPG making it the most energy-proportional subsystem.

The counters, except the ones related to ITLB and DTLB stores, have a correlation greater than 0.80 with the power consumption of the *uncore* subsystem. Our PCA revealed that four PCs (IR, NOUT, DTLBLM and LLCM) account for 83% of the variance. Our analysis reveals that the counters in Table 2 are insufficient to model the power consumption of *uncore* subsystem. We intend to look at *uncore* performance counters [2] exposed in the Sandy Bridge server platforms to uncover performance counters with better correlation in the future.

The *uncore* subsystem power remains almost constant, irrespective of the percentage of the load-level with an EP metric value of 0.17. The *uncore* subsystem has the greatest EPG, and as a result, exhibits the worst power consumption trend among the full system and RAPL domains from the perspective of energy-proportional power scaling.

Interestingly, the DRAM domain consumed a maximum average power of only 10 watts (i.e., 5 watts per socket). The performance counters related to LLC, DTLB, instructions retired and outstanding bus requests have a correlation greater than 0.91 with the power consumed by DRAM. Four PCs (LLCM, IR, DTLBLM and NOUT) retain 91% of the variance. The DRAM domain's value for the EP metric is 0.47. The SPECpower benchmark only consumes a maximum of 10 watts for the DRAM domain. Moreover, the benchmark consistently consumes its maximum power for the DRAM domain at 70% workload for all batches.

3.3.4 Profiling of the Power Plane 0 (PP0) Power

Figure 4 represents the EPG of PP0 domain. Similar to other domains, the batch size does not affect the average power consumption of the domain. All counters, except the ones related to ITLB and LLC, have strong correlation with the power profile of this domain. Four PCs (IR, NOUT, DTLBLM and DTLBSM) account for 97% of all variance. PP0 is the most energy proportional subsystem with an EP metric value of 0.86 across all batches. Interestingly, the idle power consumption of PP0 domain is 5.27 watts. The PP0 domain has the least EPG and exhibits near energy-proportional power scaling.

3.4 An Analysis of Subsystem-Level Instantaneous Power Consumption

Here we present our results for the instantaneous power profile analysis of the SPECpower benchmark. Our main goal is to visualize the opportunities for power limiting under different time windows while consuming power proportional to the load-level. We collected instantaneous power profile for three load-levels (50%, 60%, and 70%) and four different batch sizes of SPECpower using a loadable kernel module. The resolution of our power profile is limited to 5 milliseconds (ms). Our instrumentation had negligible impact on the power and performance of the SPECpower benchmark.

3.4.1 Methodology

Figures 5, 6, and 7 present the results on the instantaneous power profile analysis of SPECpower. These plots describe the opportunity for power limiting under different time windows while consuming power proportional to the load-level. The x-axes in the graphs is the resolution of time for which we profile the instantaneous power. The y-axes represents the ratio of number of instances for which the instantaneous power was greater than ideal "energy-proportional" power consumption[7] and the total number of instances for that time resolution. Henceforth, we refer to the ideal power consumption for a load-level as LL% of peak power. As an example, in Figure 5 at 50 ms resolution using 1000 as batch size, the instantaneous power was over ideal energy-proportional power consumption for 99.58, 97.25 and 93.25 percent of the total instances for 50%, 60%, and 70% load-levels, respectively.

When considering energy-proportional operation, performance is a implicit metric. Consequently, we are interested in time resolutions that will allow us to achieve energy-proportional operation with minimal or no impact on performance. These graphs can be used to determine the time window(s) useful to limit the power consumption for a certain batch size and load-level to achieve energy proportional operation. Note that we restrict our analysis to 50, 60 and 70% load-level for simplicity. However, our methodology is applicable to all batch sizes and load-levels.

3.4.2 Results

Package: Figure 5 presents the results for the instantaneous power profile analysis of the package RAPL domain. The workload consumes more than LL% of peak power over 95% of the execution time with batch size = 1000 at a resolution greater than 100 ms for all load-levels. This limits the opportunity to limit the power to achieve energy-proportional operation at time windows greater than 100 ms. However, increasing the batch size of SPECpower improves the opportunity to apply power limiting at larger time resolution. We also observe that this improvement is greater for higher load-levels. For example at 100-ms resolution, the workload consumes more than LL% of peak power for 99.83%, 99.66% and 96.16% of the total instances for 50%, 60%, and 70% load-levels with 1,000 as batch size, whereas only 96.16% to 97.16%, 92.00% of the total instances consume more than LL% of peak power when using 10,000 as batch size.

PP0: Figure 6 presents the results for the instantaneous power profile analysis of the PP0 RAPL domain. In general, PP0 provides better opportunities for power limiting

[7]For example, consuming 50% and 70% of the power consumed at 100% load-level is considered ideal energy-proportional power consumption for 50% and 70% load-levels respectively.

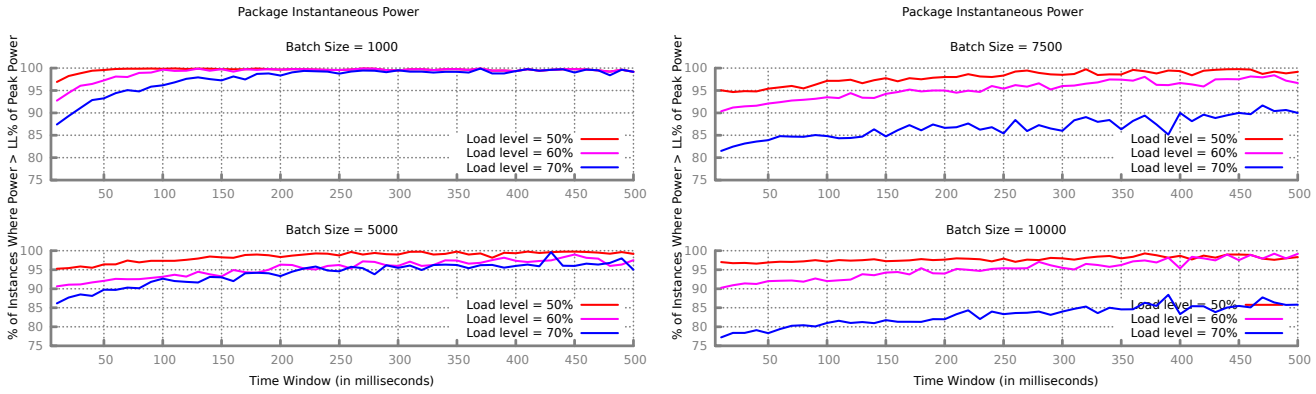

Figure 5: Instantaneous Power Analysis For Package Domain

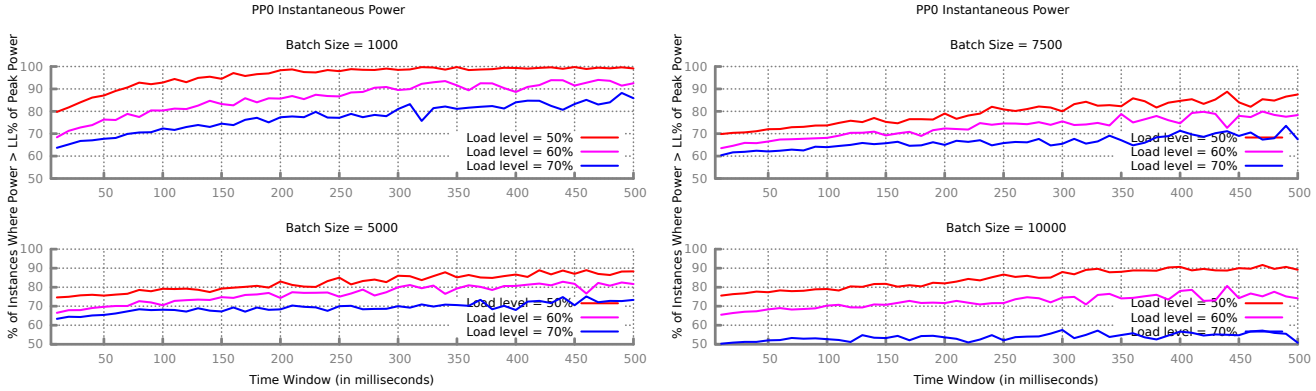

Figure 6: Instantaneous Power Analysis For PP0 Domain

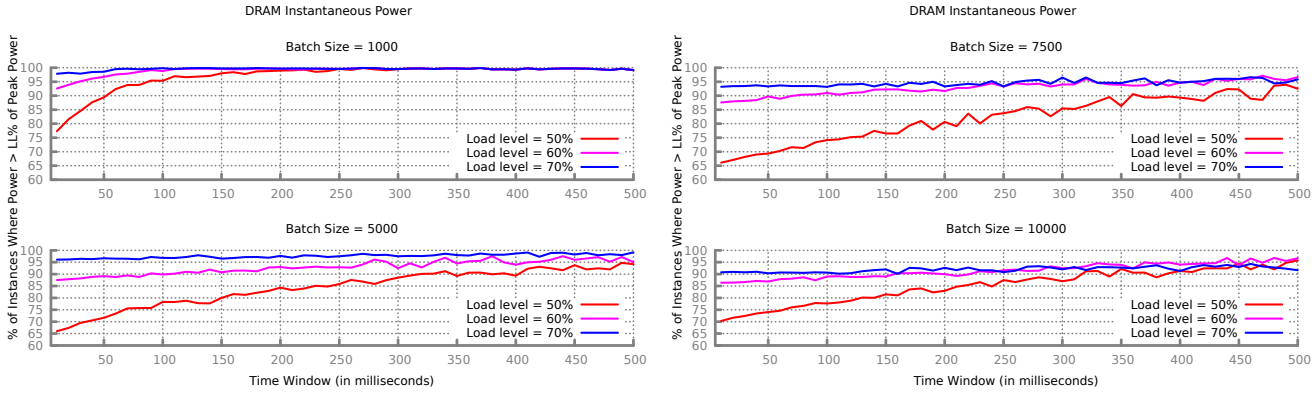

Figure 7: Instantaneous Power Analysis For DRAM Domain

at different time resolutions when compared to the package domain. This is expected as the *uncore* subsystem consumed almost constant power, irrespective of the load-level, as discussed in Section 3.3.3. Even using a batch size of 1,000, there are significant opportunities to limit power to LL% of peak power at granularities less 250 ms for 60% and 70% load-levels. Similar to the package domain, increasing the batch size improves the opportunity to limit power at larger resolution. However, the improvement is greater when compared to the package domain. For example at 100-ms resolution, the workload consumes more than LL% of peak power for 92.83%, 80.33% and 72.33% of the total instances for 50%, 60%, and 70% load-levels with 1,000 as the batch size, whereas only 79.00%, 70.33%, and 52.66% of the total instances consume more than LL% of peak power when using 10,000 as the batch size.

DRAM: Figure 7 shows the results for the instantaneous power profile analysis of the DRAM RAPL domain. In contrast to the package and PP0 RAPL domains, the opportunity for power limiting at larger time resolutions occurs

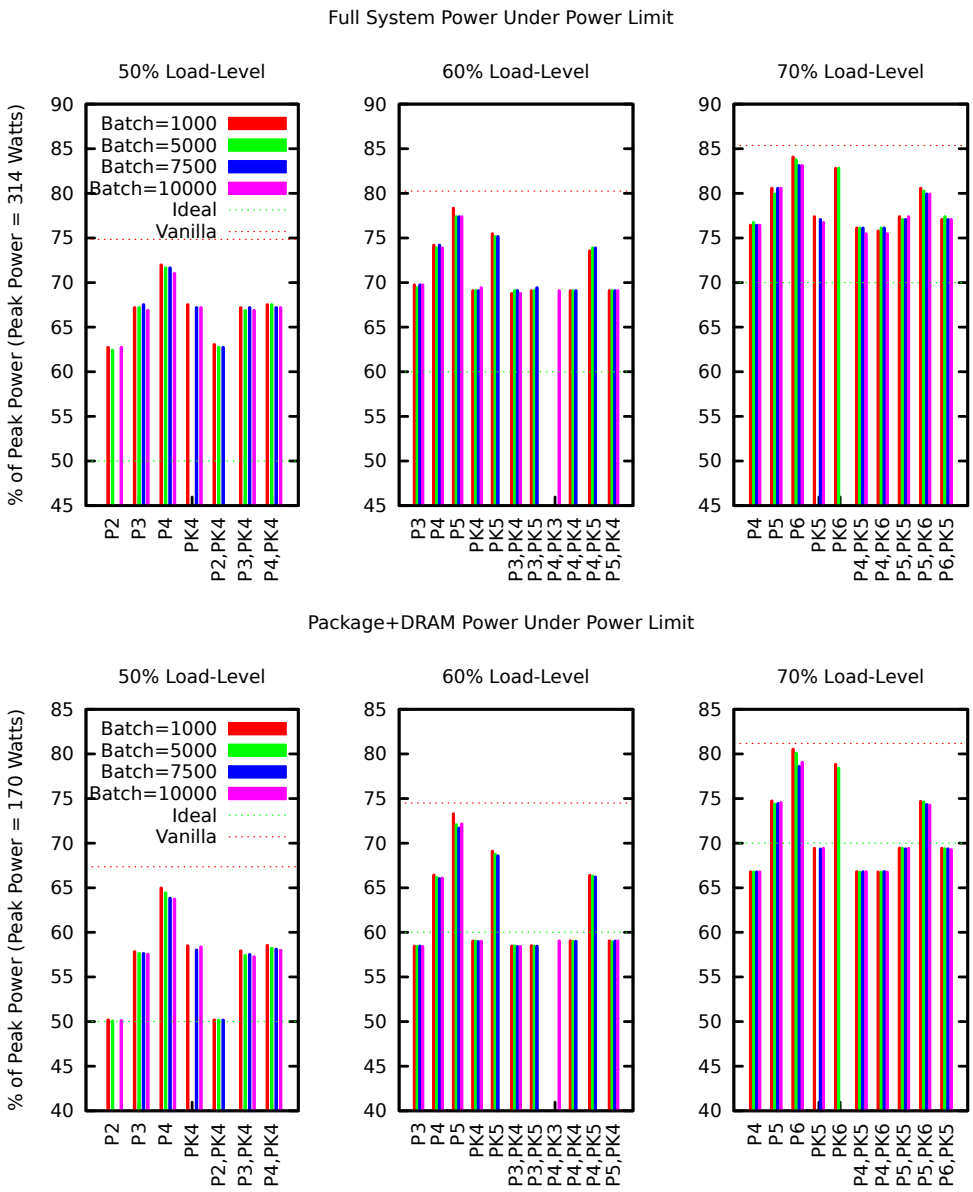

Figure 8: Power Consumption Under Power Limit

more at lower load-levels. We observe that for 60% and 70% load-levels, approximately 90% of the total instances consume more than LL% of peak power for all different batch sizes. This shows that the opportunity to achieve energy-proportional DRAM operation for these load-levels through power limiting is limited. However, 50% load-level is amenable to power limiting. Similar to the Package and PP0 domains, increasing the batch size allows power limiting even at larger time resolutions. However, the improvement is better for lower load-levels, i.e., 50% load-level has more opportunity for power limiting than 60% or 70% load-levels.

4. IMPACT OF POWER LIMITING ON SPECPOWER

In this section, we discuss the effects of power limiting on the performance and power of SPECpower. Specifically, we leverage the RAPL interfaces and analyze whether energy-proportional operation can be achieved. While we are not able to achieve energy proportionality at full system-level, RAPL power limiting can be used to improve energy efficiency without performance degradation. In addition, we are able to achieve energy-proportional operation or better at package- and memory-level (the subsystems over which we have power limiting control) for the load-levels discussed.

4.1 Methodology

We run 50%, 60%, and 70% load-levels of SPECpower under power limits by manually configuring the power limiting interfaces. We were not able to use the power limiting control for the DRAM domain for SPECpower because the maximum average DRAM power consumption (see Section 3.3.3)

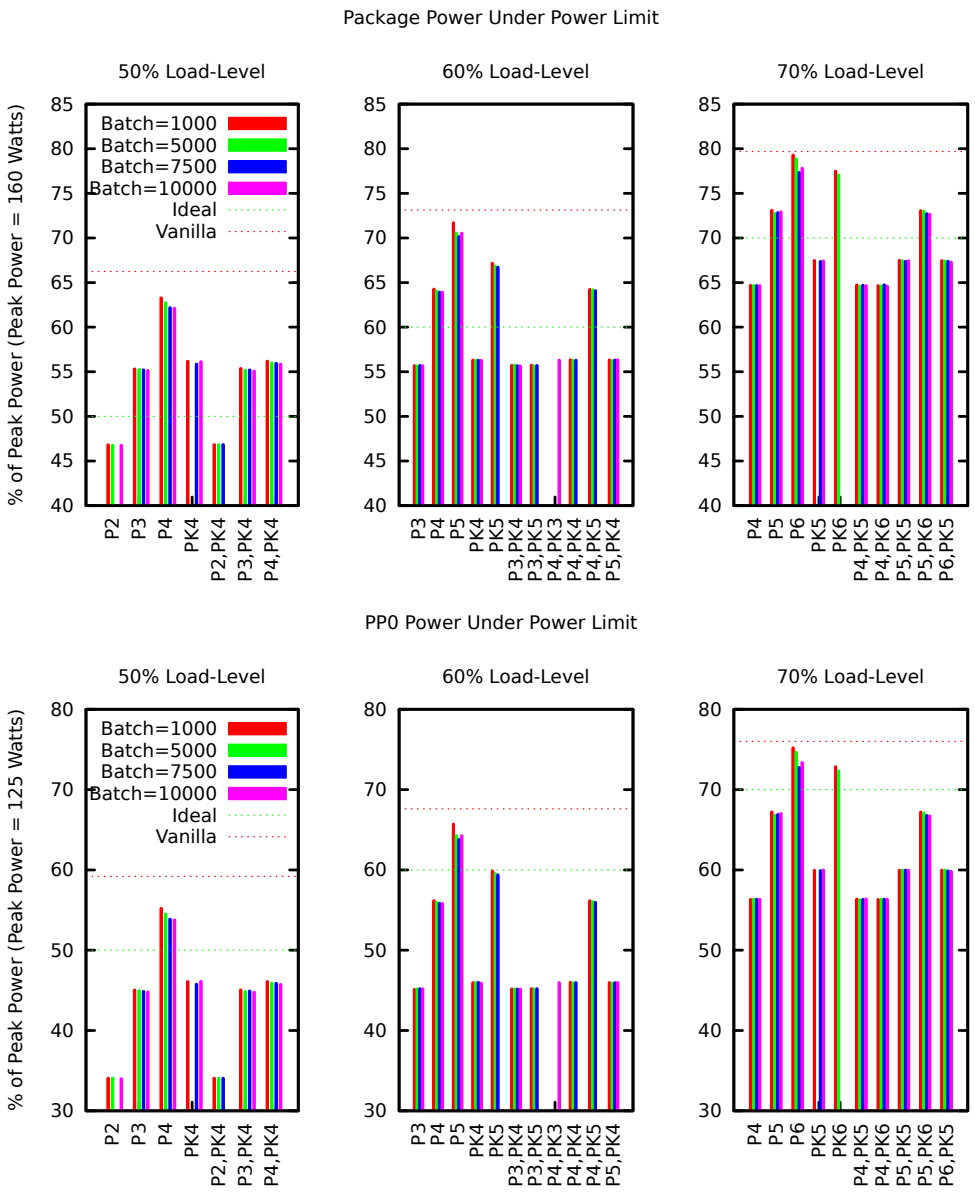

Figure 9: Power Consumption Under Power Limit

is less than the allowed minimum power consumption for the DRAM domain (see Section 2.2.4). Experiments specifying relevant power consumption limits for the DRAM domain for SPECpower resulted in no effect on power. As a result, our experiments will focus only on limiting the power consumption of the Package and PP0 RAPL domains. We manually limit the power consumption of these domains to different values of power for different load-levels, as shown in Table 3.

4.2 Impact on Power Consumption

Figures 8 and 9 show the power consumption of SPECpower under power limits for three different load-levels. We present power consumption results for the full system and three subsystems: (1) full system, (2) Package+DRAM (the domains over which we have power limiting control), (3) Package, and (4) PP0. Each plot presents only the configurations from Table 3 that achieved the particular load-level. For comparison, we also show the average power consumption of the

Table 3: Power Limit Configuration For Each Load-Level (In % of Peak Power For That Domain)

Load-Level/Domain	PP0	Package
50	20, 30, 40, 50(P1, P2, P3, P4)	25, 30, 40, 50(PK1, PK2, PK3, PK4)
60	30, 40, 50, 60(P2, P3, P4, P5)	30, 40, 50, 60(PK2, PK3, PK4, PK5)
70	40, 50, 60, 70(P3, P4, P5, P6)	40, 50, 60, 70(PK3, PK4, PK5, PK6)

subsystem without power bounds (vanilla – red dotted line) and energy-proportional power (ideal – green dotted line). All the results presented in Figures 8 and 9 use the smallest time window for power limiting (i.e., 976 microseconds).

We observe that power limiting does not allow us to enable energy-proportional operation at the full system level. We would like to emphasize that the system consumed 37.2% of peak power (i.e., 117 watts) even when idling. However, we achieved moderate power savings at the full system level *without* any performance degradation by using power limiting. We also observe that the power saving increases with decreased load-level. For example, the best full system power saving achieved at 50% and 70% load-level are 13% and 9% over the vanilla runs of SPECpower, respectively.

We achieved energy-proportional operation at the PKG+ DRAM, PKG, and PP0 RAPL domain levels for all load-levels. Power limiting in the PP0 RAPL domain is the best possible mechanism to achieve energy-proportional operation. Interestingly, the combination of a PP0 power limit and package power limit always consumed approximately the same power as a corresponding run with only the PP0 power limit. Our power-limiting experiments with only the package domain revealed that approximately 98.5% of the power savings came from the PP0 domain for all load-levels. The reason for such power profiles was due to the fact that the *uncore* consumes almost constant power, irrespective of the load-level (see Section 3.3.3). As such, we conclude that better mechanisms are needed to control the power consumption of the *uncore* subsystem [2].

4.3 Impact on Response Time

The response time in workloads such as SPECpower can greatly impact user experience. In this section, we measure the impact of power limiting on per-transaction response time. The response time can vary widely even if the throughput (i.e., load-level) is maintained. The arrival time, wait time, and total response time of batches in a run can be logged by setting the *input.scheduler.log_arrival_rates* to true. We are interested in the total response time (wait + execution time) as it is directly related to the performance seen by a user.

Figure 10 shows the average slowdown on per-transaction response time of three different load-levels and four different batch sizes under a power limit. The slowdown is defined as the ratio of response time under the power limit and the response time of a vanilla run. As expected, the response time varies widely within a given load-level. PP0 power limiting gives the best power-performance trade-off; the response time is directly correlated to batch size for all load-levels. For larger batch sizes, the response time for SPECpower decreases. The response time with batch size of 1,000 worsens as we use package power limiting. We also observe that power limiting can have drastic effects on the response time for different batch sizes. Determining the *right* power limit to achieve a certain load-level with acceptable per-transaction response time is an avenue for future work.

4.4 Impact on Energy Efficiency

As discussed earlier, SPECpower is a power and performance benchmark with ssj_ops/watt (i.e., performance-to-power ratio) as the main metric of interest. Figure 11 depicts the energy efficiency improvement under power limits. The improvement is computed as the ratio of the difference

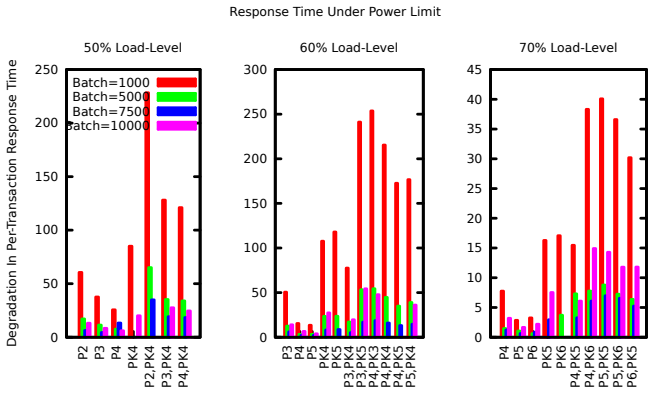

Figure 10: Response Time Under Power Limit

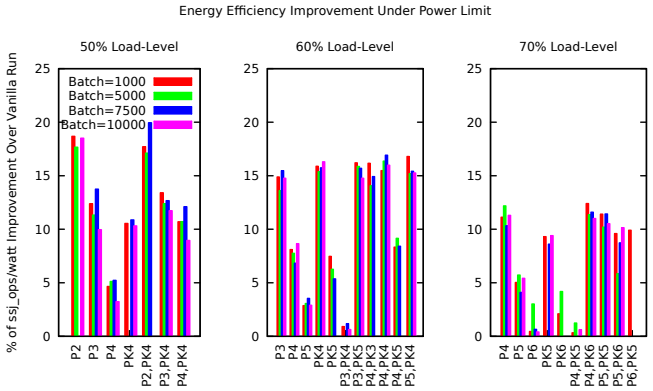

Figure 11: Energy Efficiency Improvement Under Power Limit

between the energy efficiency under a power limit and vanilla run (i.e., no power limit) over the efficiency in a vanilla run. We observe that the percentage gain in energy efficiency decreases with increase in load-level. Power limiting enables us to achieve a maximum energy efficiency improvement of 20%, 16%, and 12% at 50%, 60%, and 70% load-levels, respectively. These results show that PP0 power limiting (in isolation) is the best mechanism to achieve better energy efficiency.

4.5 Impact of a Power-Limiting Time Window on Response Time

Figure 12 shows the impact of a power-limiting time window on the per-transaction response time for running SPECpower with a batch size of 5,000 and setting a power limit for PP0 RAPL domain. We show only one batch size and load-level for simplicity. The rest of the batch sizes and load-levels showed similar trends. We emphasize that the average power consumption of SPECpower remains approximately the same for a specified power limit with different power-limiting time windows. There is a huge improvement moving from a ∼2ms to a ∼4ms time window. In general, increasing the time window improves the response time of SPECpower, though this improvement asymptotes as the size of the window continues to increase.

Figure 12: Impact of a Power-Limiting Time Window on Response Time

5. RELATED WORK

We classify the related work in three categories: (1) energy-proportional operation for enterprise-class workloads, (2) characterization and power analysis of the SPECpower benchmark, and (3) power limiting.

5.1 Energy-Proportional Operation For Enterprise Class Workloads

Our work is most related to the work of Meisner et al. [22]. They characterize online data-intensive services (OLDI) to identify opportunities for power management, design a framework that predicts the performance of OLDI workloads and investigate the power and performance trade-offs using the framework. Our work leverages Intel's RAPL interfaces to characterize the power consumption of SPECpower and analyze whether it is possible to achieve energy-proportional operation on a system.

Fan et al. [13] investigate the benefits of energy-proportional systems in improving the efficiency of power provisioning using their power models. They provide evidence that energy-proportional systems will enable improved power-capping at the data-center level. In contrast, we look at leveraging the *power-capping* mechanism to achieve energy-proportional operation for SPECpower.

Tolia et al. [28] proposed that by migrating workloads from under-utilized systems to other systems and turning the under-utilized systems off, energy proportionality can be approximated at an ensemble-level (i.e., for a group of nodes or rack-level). They used virtual machine (VM) migration as a software mechanism to move workloads off of under-utilized systems. In this paper, we use user-defined and hardware-enforced power limiting to achieve energy-proportional systems at the node-level.

5.2 Charecterization and Power Analysis of SPECpower

Fanara et al. [14] and Lange [20] comment on the design and development of the SPECpower benchmark. An overview of the workload, its behavior, and detailed characterization of the SPECpower benchmark under different load-levels is described in [17]. Our goal in this paper is to characterize SPECpower to provide insights into the opportunities for power management for an enterprise-class server workload.

Hsu et al. [19] defined accurate and portable power models that capture the linearity of the power curve under different load-levels of SPECpower. Varsamopoulos et al. [29] proposed the idle-to-power ratio (IDR) and linear deviation ratio (LDR) metrics, analyzed a large number of SPECpower results, and provided insights into server provisioning. Similarly, Ryckbosch et al. [26] proposed the EP metric to quantify the energy proportionality of the system. In this paper, for the first time, we quantify the energy proportionality at a subsystem-level using the on-chip energy meters exposed through RAPL. We also characterize the instantaneous power profile at different load levels of SPECpower.

5.3 Power Limiting

Several mechanisms to *cap* the power consumption of the system have been studied [11, 16]. However, we study the use of RAPL power limiting which is hardware-enforced in this paper. David et al. [12] proposed RAPL and evaluated RAPL for the memory sub-system. They present a model that accurately predicts the power consumed by the DIMMs and use RAPL to *cap* the power consumption. Rountree et al. [24] use RAPL power limiting to study the behavior of performance for benchmarks in the NAS parallel benchmark suite. Specifically, they are interested in the performance of various compute nodes under a power bound. Weaver et al. [30] have have exposed RAPL energy meters through PAPI. We use RAPL interfaces to achieve energy-proportional operation for SPECpower benchmark and to the best of our knowledge, there is no previous study on using RAPL interfaces for enterprise class server workloads.

6. CONCLUSION

The management of power and energy is a key issue for data centers. Efficient power management of enterprise-class server workloads have the potential to greatly reduce energy-related costs and facilitate efficient power provisioning.

Energy proportionality holds the potential to significantly improve the energy efficiency of data centers. Consequently, in this paper, we investigate the potential of achieving energy proportionality for SPECpower benchmarks using RAPL interfaces. Our study sheds light on the mechanisms for power management of enterprise-class server workloads and the efficacy of RAPL interfaces. We identify the least and most energy-proportional subsystem using the on-chip energy meters. Performance counter traces are collected to identify which events have strong correlation with power consumption at a subsystem-level. We also characterize the instantaneous power profile of SPECpower and identify the time resolutions at which there is opportunity to limit the power consumption of the benchmark. We use RAPL power limiting and show that we are able to achieve energy-proportional operation (or better) at the subsystem-level.

7. ACKNOWLEDGMENTS

The authors would like to thank Ashwin Aji, Vedvyas Duggirala, and Hari Pyla for their consultation and the anonymous reviewers for their feedback. This work was supported in part by a gift and equipment donation from Supermicro.

8. REFERENCES

[1] Intel 64 and IA-32 Software Developer Manuals - Volume 3. Available at `www.intel.com/content/www/us/en/processors/architectures-software-developer-manuals.html`.

[2] Intel Xeon Processor E5-2600 Product Family Uncore Performance Monitoring Guide. Available at `http://www.intel.com/content/dam/www/public/us/en/documents/design-guides/xeon-e5-2600-uncore-guide.pdf`.

[3] SPECjbb Benchmark. Available at `http://www.spec.org/jbb2005/`.

[4] SPECpower Benchmark. Available at `http://www.spec.org/power_ssj2008`.

[5] SPECpower Benchmark – Benchmarking Methodology. Available at `http://www.spec.org/power/docs/SPEC-Power_and_Performance_Methodology.pdf`.

[6] SPECpower Benchmark – CCS Design Document. Available at `http://www.spec.org/power/docs/SPECpower_ssj2008-Design_ccs.pdf`.

[7] SPECpower Benchmark – Design Overview. Available at `http://www.spec.org/power/docs/SPECpower_ssj2008-Design_Overview.pdf`.

[8] SPECpower Benchmark – Run Rules. Available at `http://www.spec.org/power/docs/SPECpower_ssj2008-Run_Reporting_Rules.html`.

[9] SPECpower Benchmark – SSJ Workload Design Document. Available at `http://www.spec.org/power/docs/SPECpower_ssj2008-Design_ssj.pdf`.

[10] L. A. Barroso and U. Hölzle. The Case for Energy-Proportional Computing. *IEEE Computer*, 40(12):33–37, 2007.

[11] R. Cochran, C. Hankendi, A. K. Coskun, and S. Reda. Pack & Cap: Adaptive DVFS and Thread Packing Under Power Caps. In *Proc. of the Int'l Symp. on Microarchitecture*, MICRO-44, pages 175–185, 2011.

[12] H. David, E. Gorbatov, U. R. Hanebutte, R. Khanna, and C. Le. RAPL: Memory Power Estimation And Capping. In *Proceedings of the International Symposium on Low Power Electronics and Design*, ISLPED, pages 189–194, 2010.

[13] X. Fan, W.-D. Weber, and L. A. Barroso. Power Provisioning For a Warehouse-Sized Computer. In *Proceedings of the International Symposium on Computer Architecture*, ISCA, 2007.

[14] A. Fanara, E. Haines, and A. Howard. The State of Energy and Performance Benchmarking for Enterprise Servers. In R. Nambiar and M. Poess, editors, *In Performance Evaluation and Benchmarking*, pages 52–66. Springer-Verlag, 2009.

[15] M. Ferdman, A. Adileh, O. Kocberber, S. Volos, M. Alisafaee, D. Jevdjic, C. Kaynak, A. D. Popescu, A. Ailamaki, and B. Falsafi. Clearing the Clouds: A Study of Emerging Scale-out Workloads on Modern Hardware. In *Proceedings of International Conference on Architectural Support for Programming Languages and Operating Systems*, ASPLOS, pages 37–48, 2012.

[16] A. Gandhi, M. Harchol-Balter, R. Das, J. Kephart, and C. Lefurgy. Power Capping Via Forced Idleness. In *Proceedings of Workshop on Energy-Efficient Design*, WEED, 2009.

[17] L. Gray, A. Kumar, and H. Li. Workload Characterization of the SPECpower_ssj2008 Benchmark. In *Performance Evaluation: Metrics, Models and Benchmarks*, Lecture Notes in Computer Science, pages 262–282. 2008.

[18] C. Hsu and W. Feng. A Power-Aware Run-Time System for High-Performance Computing. In *Proceedings of the SC Conference*, 2005.

[19] C. Hsu and S. W. Poole. Power signature Analysis of the SPECpower_ssj2008 Benchmark. In *Proc. of the Int'l Symp. on Performance Analysis of Systems and Software*, ISPASS, pages 227–236, 2011.

[20] K. Lange. Identifying Shades of Green: The SPECpower Benchmarks. *Computer*, 42(3):95–97, Mar. 2009.

[21] D. Meisner, B. T. Gold, and T. F. Wenisch. PowerNap: Eliminating Server Idle Power. In *Proceedings of International Conference on Architectural Support for Programming Languages and Operating Systems*, ASPLOS, 2009.

[22] D. Meisner, C. M. Sadler, L. A. Barroso, W.-D. Weber, and T. F. Wenisch. Power Management of Online Data-Intensive Services. In *Proceedings of the International Symposium on Computer Architecture*, ISCA, pages 319–330, 2011.

[23] S. Rivoire. Models and Metrics for Energy-Efficient Computer Systems. *Ph.D. Dissertation, Department of Electrical Engineering, Stanford*, 2008.

[24] B. Rountree, D. Ahn, B. de Supinski, D. Lowenthal, and M. Schulz. Beyond DVFS: A First Look at Performance Under a Hardware-Enforced Power Bound. In *Proc. of the Int'l Parallel and Distributed Processing Symp. Workshops and PhD Forum*, IPDPSW, pages 947–953, 2012.

[25] B. Rountree, D. K. Lownenthal, B. R. de Supinski, M. Schulz, V. W. Freeh, and T. Bletsch. Adagio: Making DVS Practical For Complex HPC Applications. In *Proc. of the Int'l Conf. on Supercomputing*, ICS, pages 460–469, 2009.

[26] F. Ryckbosch, S. Polfliet, and L. Eeckhout. Trends in Server Energy Proportionality. *IEEE Computer*, (9):69–72, 2011.

[27] D. C. Snowdon, S. M. Petters, and G. Heiser. Accurate On-line Prediction of Processor and Memory Energy Usage Under Voltage Scaling. In *Proceedings of the International Conference on Embedded Software*, EMSOFT, pages 84–93, 2007.

[28] N. Tolia, Z. Wang, M. Marwah, C. Bash, P. Ranganathan, and X. Zhu. Delivering Energy Proportionality with Non Energy-Proportional Systems: Optimizing the Ensemble. In *Proceedings of the Conference On Power Aware Computing and Systems*, HotPower. USENIX Association, 2008.

[29] G. Varsamopoulos, Z. Abbasi, and S. Gupta. Trends and Effects of Energy Proportionality on Server Provisioning in Data Centers. In *Proceedings of the International Conference on High Performance Computing*, HiPC, pages 1–11, 2010.

[30] V. Weaver, M. Johnson, K. Kasichayanula, J. Ralph, P. Luszczek, D. Terpstra, and S. Moore. Measuring Energy and Power with PAPI. In *Int'l Workshop on Power-Aware Systems and Architectures*, 2012.