Application-aware Power Saving for Online Transaction Processing using Dynamic Voltage and Frequency Scaling in a Multicore Environment

Yuto HAYAMIZU, Kazuo GODA, Miyuki NAKANO, and Masaru KITSUREGAWA

Institute of Industrial Science, the University of Tokyo
4-6-1 Komaba, Meguro-ku, Tokyo 153-8505 Japan
`haya@tkl.iis.u-tokyo.ac.jp`

Abstract. Power consumption in data centers has been growing remarkably recent years, and power saving of their servers is essential. For power saving of these servers, power saving of an online transaction processing (OLTP) systems, which are major applications in data centers, is important. The OLTP system consumes relatively large amount of power because it is often equipped with a lot of computing and storage resources. Its power saving is difficult because it is required to meet a service level agreement (SLA), and few power saving technologies have been proposed so far.
In this paper, we proposed an application-aware power saving for OLTP in a multicore environment. Our proposed methodology aims to save power consumption of OLTP systems by dynamically scaling the operating frequency of processors based on response time observation. Response time is often an important metric of SLA. Application-aware power saving enables power saving in such systems subject to SLAs. In our experimental evaluations using industrial standard benchmark TPC-C and real server workloads, 7.6% of total power consumption was saved. This reduction corresponds to 1000kJ a day in a typical entry level server.

Keywords: database system, multicore, online transaction processing, power saving, application aware, service level agreement, dynamic voltage and frequency scaling

1 Introduction

Power consumption in data centers has been growing remarkably in recent years. The U.S. Environmental Protection Agency reported that power consumption of data centers in the United States doubled from 2000 to 2006, and would double again in another five years[4]. This power consumption growth causes not only energy cost increases but also impediments to innovation due to shortage of power supply.

EMERSON reported that servers in a normal data center account for 44% of total power consumption, and their cooling systems account for 38% of the total[15]. Power saving of servers is essential for saving power at data centers. Power saving of servers results in less power distribution equipment, AC/DC converters, UPSs, and so on. This also enables reduction of cooling systems. In other words, power saving of servers leads to total power saving at data centers.

In data centers, database servers play a central role in managing and processing much information. Online transaction processing (OLTP) is a major application running on database servers. Generally, OLTP database servers are required to achieve high throughput and low latency. So a lot of computing and storage resources are put in these servers, resulting in considerable power consumption. In addition, OLTP database servers cannot be turned off because it would directly lead to loss of business opportunities. Power saving of OLTP servers is an essential challenge. Poess et. al. presented a basic trend analysis of power consumption and energy efficiency of registered top systems with the TPC-C record, the industrial standard benchmark of OLTP, in [12], but few effective power saving techniques for OLTP have been proposed so far.

In this paper, we propose an application-aware power saving for OLTP in a multicore environment. Our proposed methodology aims to save power consumption of OLTP database servers while complying with service level agreements (SLA) by dynamically scaling the operating frequency of processors based on response time observation. Response time is often an important performance metric of real OLTP systems. Application-aware power saving can enable power saving in such systems subject to SLAs. In all our experiments, the TPC-C benchmark was used as an OLTP application. As a result of evaluations with actual server workloads, 7.6% of total power consumption was saved.

The rest of this paper is organized as follows: Section 2 introduces OLTP and presents a basic idea of application-aware power saving. Section 3 details our method of application-aware power saving for OLTP. Section 4 describes an experimental environment and basic performance/power measurements. Section 5 evaluates our proposed methodology with artificially synthesized workloads and traces of real servers. Section 6 surveys related work and Section 7 gives concluding remarks.

2 Power Saving in Online Transaction Processing

2.1 Online Transaction Processing

Online transaction processing (OLTP) is *the* infrastructure technology for various social and economic services such as online banking, e-commerce and e-trading. Typical OLTP systems need to handle a vast number of small transactions simultaneously. Performance is a top priority in system design and operation. Database systems have to achieve high throughput for handling these transactions. Short response time is also important in practical OLTP servers. In many cases, the system is required to respond to each request in a prescribed amount of time.

These days, mid-range or even entry-level servers are configured with multicore processors and large amounts of memory in order to achieve high performance by exploiting multicore processors with multi threads and by caching databases in the main memory. More OLTP systems have come to be built in such environments. Power consumption of processors accounts for a large part of the total power. This is typical for x86 architecture processors, which are popular in the volume server market. Saving power consumption of the processor is effective for total power saving. Therefore, we focused on processor power saving in this paper.

2.2 Application-aware Power Saving

Balancing performance and power saving is an essential problem for making transaction processing systems energy-efficient. High performance (high throughput and low latency) is usually a primary requirement. Design interest is in how to control processor throttles for minimizing power utilization under a given performance requirement.

There can be a design spectrum for power saving of transaction processing systems. In this paper, let us discuss an *application-aware* approach, where the system can observe behavior information of running applications and then utilize it for manipulating power modes of processors. Recent transaction processing is becoming more mission critical. More systems have come to operate under a service level agreement (SLA), which strictly specifies performance requirements. Such requirements are mostly given at application level. Observing a system at application level is a must for its administration. Similarly, the application-aware approach looks reasonable for power saving. Suppose that we are trying to keep a given restriction of transaction response time and at the same time saving power as much as possible. Measuring transaction response time directly from running applications is a straightforward solution. Although design alternatives that are application-independent may be able to give some power saving, the application-aware approach potentially provides more efficient power/performance balancing. To our knowledge, an attempt of this approach has not been studied for transaction processing systems in literature. In this paper, we clarify the approach's benefits using a real experiment system and a de-facto standard benchmark of transaction processing.

One design concern about the application-aware approach is how to build and tune control mechanism for every specific application. Further investigation is necessary, but we are expecting the possibility of extending recent system management frameworks, which many enterprises use to manage performance and resource utilization of multiple applications in consolidated systems.

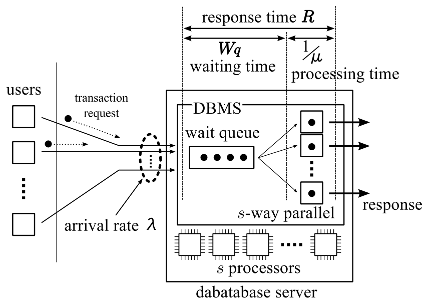


Fig. 1. A transaction processing model of a database server

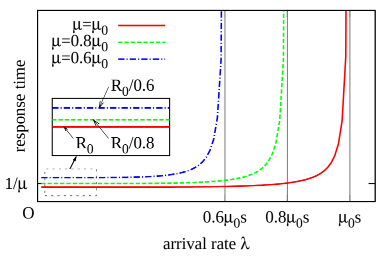


Fig. 2. Shapes of response time R curves of the model

3 Application-aware Dynamic Scaling for Online Transaction Processing

3.1 Dynamic Voltage and Frequency Scaling

Many processors available in the market have *Dynamic Voltage and Frequency Scaling* (DVFS), which is a function to change the operating frequency and the voltage dynamically. DVFS gives opportunities for changing operating frequency and voltage when a processor is not fully busy, thus potentially reducing the power consumption. For example, the Intel Xeon processor has a DVFS mechanism named Intel SpeedStep Technology. A system program such as an OS kernel can directly control the operating frequency by writing the desired frequency value to a special control register.

Power consumption of a processor can be decomposed to two parts: idle power P_{idle}, and active power P_{active}. P_{idle} is approximately a fixed power consumption which is independent from operating frequency and voltage. If a processor is fully idle, the total power consumption is equal to P_{idle}. P_{active} is an incremented part due to operation loads. P_{active} is known to be in the relation of $P_{\text{active}} \propto fV^2$, where f is the operating frequency and V is the operating voltage. Our preliminary experiments affirmed that the operating voltage of the Intel Xeon processor is mostly constant regardless of the operating frequency, so power consumption of our system is proportional to the operating frequency.

3.2 Application-aware Dynamic Frequency Scaling for Online Transaction Processing

In order to design an algorithm for controlling the operating frequency of processors, we modeled an OLTP database server to understand the relationship between the operating frequency and OLTP application performance.

Fig. 1 illustrates a model of transaction processing. In an OLTP application, multiple users send transaction requests to a database server, and they are processed by a database management system (DBMS). Given s is the number of processors in a server [1], the DBMS can process s transaction in parallel, so it can be considered as a wait queue with s windows. P_0 (the probability of having no waiting or on-going transactions) , W_q (average queuing time) and R (average response time) are described as follows. Note that λ and μ denote respectively transaction arrival rate, and average throughput of one processor in a database server.

$$P_0 = \frac{1}{\sum_{k=0}^{s} \frac{\rho^k}{k!} + \frac{\rho^{s+1}}{s!(s-\rho)}} \tag{1}$$

$$W_q = \frac{s\rho^{s+1}}{s!(s-\rho)^2} P_0 \tag{2}$$

$$R = W_q + \frac{1}{\mu} = \frac{s\rho^{s+1}}{s!(s-\rho)^2} P_0 + \frac{1}{\mu} \tag{3}$$

where $\rho = \lambda/\mu$.

In this model, average throughput μ is proportional to the operating frequency. To examine the effect of operating frequency change on performance, we plotted the average response time R for each $\mu = \mu_0, 0.8\mu_0, 0.6\mu_0$ in Fig. 2. For every μ, R is equal to $1/\mu$ at $\lambda = 0$, monotonically increases according to λ, and diverges when λ is equal to μ. That is to say that the maximum throughput of the database server is μs, and response time gets very long when λ is close enough to μs.

In the area marked with a rectangle in Fig. 2, the transaction arrival rate and the system usage are relatively low. In these cases, W_q is almost equal to 0, therefore $R = 1/\mu$. Since μ is proportional to operating frequency, if we throttle frequency from f to αf ($\alpha < 1$), response time is increased from R to R/α. Given that the response time R_0 at $\mu = \mu_0$, when $\mu = 0.8\mu_0, 0.6\mu_0$ response times are respectively $R = R_0/0.8, R_0/0.6$.

Considering the above, we designed the following frequency control algorithm for application-aware power saving.

- Monitor the average response time at a specified sampling rate:
  - When average response time is equal to or greater than an up-provisioning threshold R_{up}, raise the operating frequency by one level.
  - When average response time is equal to or less than a down-provisioning threshold R_{down}, drop the operating frequency by one level.

As described above, when the response time increases to R_{up}, then this algorithm adjusts the operating frequency based on the change in response time. Thresholds $R_{\text{up}}, R_{\text{down}}$ have hysteresis because switching of the operating frequency happens frequently and causes performance loss if $R_{\text{up}} = R_{\text{down}}$ and average response time R_{avg} is close to R_{up}. By putting a moderate split between $R_{\text{up}}, R_{\text{down}}$, this frequency flapping could be suppressed.

Though we can make various application-aware DVFS algorithms other than this one, comparing them should be future work because we have focused on clarifying the efficacy of application-aware power saving in this paper.

[1] In this context, a processor corresponds to a core of a multicore processor.

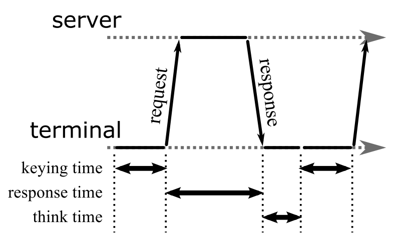


Fig. 3. Flow diagram of a TPC-C terminal

4 Experimental Environment and Basic Measurements

We built an experimental environment to verify the effectiveness of our proposed application-aware power saving on real database servers using TPC-C benchmark. Presented in this section are an overview of the environment and the basic performance measurements thereof.

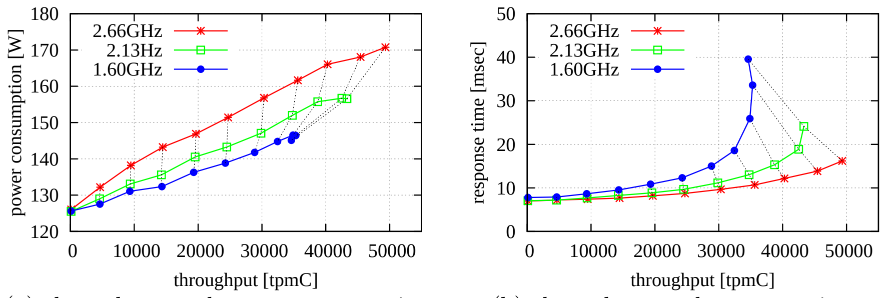
4.1 Experimental Environment

We conducted all our experiments with a database server Dell PowerEdge R510 equipped with two Intel Xeon X5550 2.67GHz processors and 48GB of memory (DDR3 8GB RDIMM $\times$ 6). The Intel Xeon X5550 processor has nine levels of operating frequencies from 1.60GHz to 2.66GHz. We attached multiple clamp meters to power supply cables of the server and connected them to a data logger (Hioki 2332-20 Power Meter Module). This combination could measure and record total power consumption of the server at the highest sampling rate, 1 hertz. TPC-C terminals ran on a single machine, Dell PowerEdge R900, equipped with four Intel Xeon X7460 2.66GHz processors and 128GB of memory (DDR2 4GB FB-FIMM $\times$ 32).

As a DBMS, MySQL 5.1.41 with InnoDB storage engine was used on Linux kernel 2.6.18. To emulate recently popular configurations, where the large memory is often used for intensively caching databases and reducing IOs, data files of InnoDB were placed on a `tmpfs` memory file system, and allocated enough buffer pool to cache all data. The kernel contains a driver for a DVFS function of the processor, `cpufreq`, and it was used for adjusting the operating frequency of the processor in our experiments.

TPC-C [16], the industrial standard benchmark, was used as the OLTP application with parameters compliant with TPC-C specification version 5.11 except as specified below. For each experiment, the database was initialized with 100 warehouses (a scale factor of TPC-C). The number of terminals (virtual users) was set to 100 and each terminal was attached to a different warehouse.

The transaction arrival rate was throttled by adjusting the length of sleep operations called *keying time* and *think time*. As shown in Fig. 3, the TPC-C standard specifies that a terminal must repeat the following operations: 1. sleep



(a) throughput and power consumption (b) throughput and response time

Fig. 4. The results of basic performance measurements

for a given amount of keying time, 2. send a transaction request, 3. receive a response, and 4. sleep for a given amount of think time. A shorter keying time and think time means a higher transaction arrival rate. In our experiments, each terminal was invoked with arranged keying time and think time such that the system could produce an aimed transaction arrival rate in total. The relationship between keying time, think time and resulted arrival rate was obtained by preliminary experiments.

The parameters of the proposed DVFS control algorithm were also decided by preliminary experiments: $R_{\text{up}} = 20$[msec], $R_{\text{down}} = 15$[msec]. The sampling rate of average response time was set to 1 second.

4.2 Basic Measurements

In order to understand the performance and power consumption of our system, we conducted basic measurements of TPC-C on our environment with combinations of the three operating frequencies (2.66, 2.13, 1.60GHz), and the eleven arrival rates (0, 5000, 10000, 15000, ..., 50000tpmC). For each configuration, TPC-C was executed for 20 minutes and power consumption, throughput and response time were measured. Every reported value is an average during a steady state, which is a time range excluding ramp-up and ramp-down.

The results are shown in Fig. 4. Data points connected with a dotted line are of the same transaction arrival rate. As shown in Fig. 4(a), for each transaction arrival rate under 30000tpmC, there were weak dependencies between throughput and operating frequency. However in the high arrival rate region, throughput values got lower. See Fig. 4(b), which shows that, in the same region, response time values got remarkably longer due to the reduced operating frequency. This means that a response time increase could be an indicator of throughput degradation, and our proposed DVFS control algorithm could prevent the degradation with this indicator.

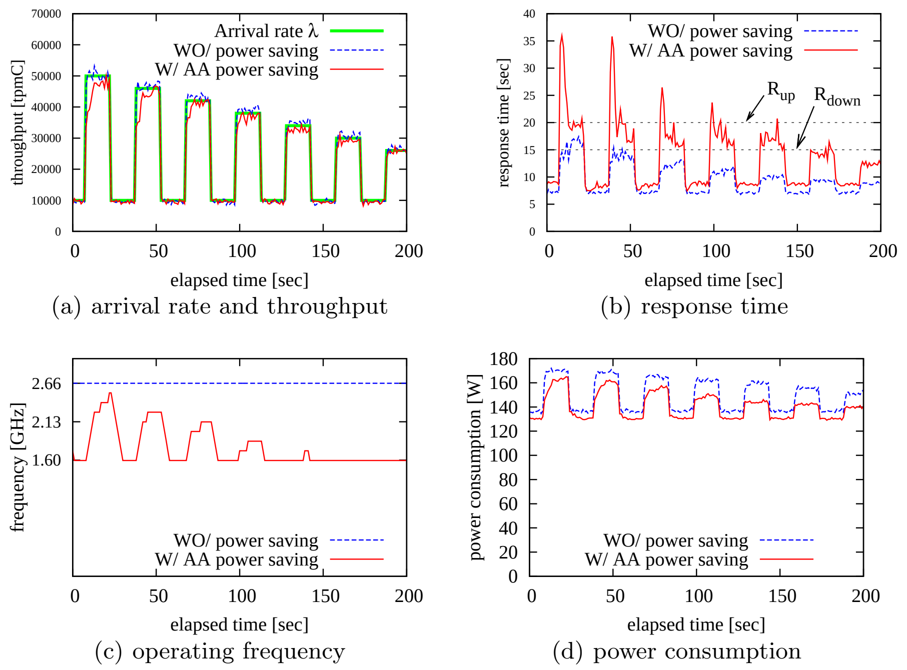


Fig. 5. Experimental results with the synthetic workload

5 Evaluation of Application-aware Power Saving

5.1 Synthetic Workload

We examined the behavior of our proposed DVFS control algorithm in response to changes of transaction arrival rate. We injected a synthetic workload as shown in Fig. 5(a): the transaction arrival rate followed a 30-second-period square wave, of which peak values were decreased gradually. Measurements were done in two cases: “WO/ power saving” operating at the fixed default frequency (2.66GHz) and “W/ AA power saving” dynamically controlling the operating frequency using our algorithm.

Results are plotted in Fig. 5. The proposed method could dynamically scale the operational frequency based on the observation of transaction response time, and save power consumption as much as possible. When the transaction arrival rate was low at around 10,000 tpmC, the frequency was set to the lowest at 1.60GHz for saving power as much as possible because this frequency could keep the response time under R_{down}. But, the first five waves were so high at peak that the operational frequency of 1.60GHz made the response time higher than R_{up}. The proposed method could then dynamically raise up the frequency level step by step and finally brought the response time back under R_{up}. Similarly, it could step down the frequency when the transaction rate dropped down back. See the first wave (the steepest case), where the proposal needed only five seconds for erasing performance degradation due to the workload fluctuation. We could further optimize the configuration such as R_{up} and R_{down}, but this penalty looks acceptable in many cases. For the sixth and later waves, the peak transaction rate was not so high to reach R_{up}. Thus the proposed algorithm could avoid raising up the frequency for saving power as much as possible. Viewing the experiment overall, we verified that the proposed method could save power of the OLTP server with small performance penalty in comparison with the default configuration that is performance-oriented without power saving.

5.2 Real-world Workload

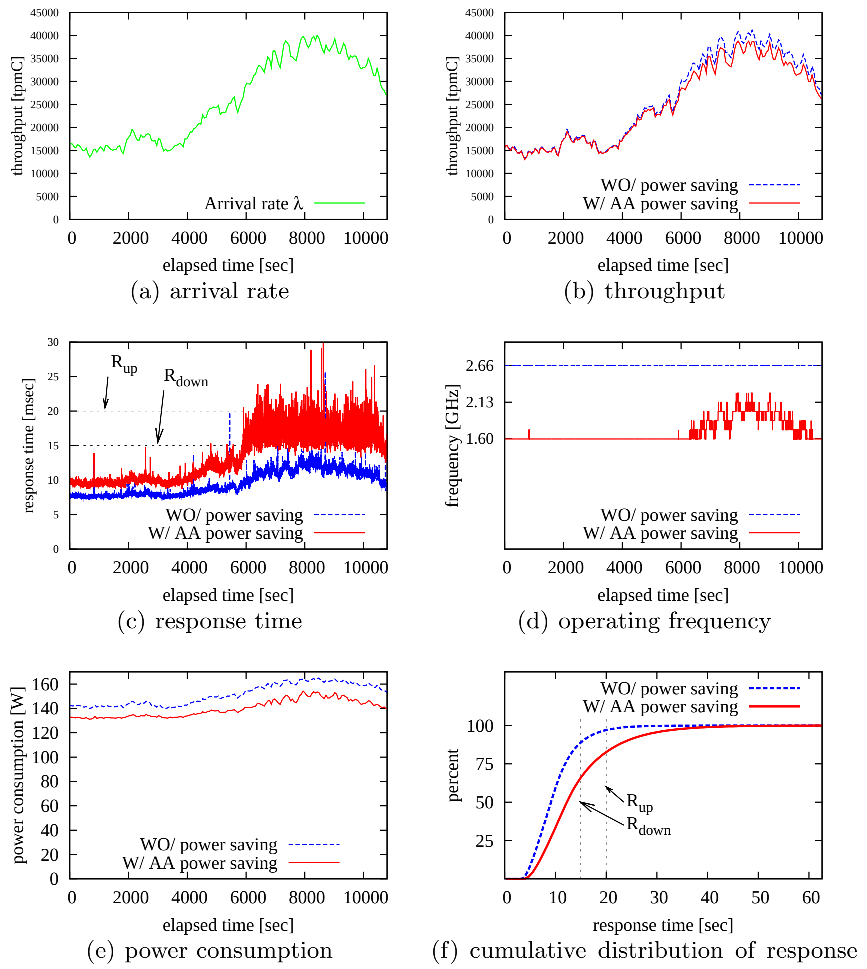
We evaluated the efficacy of our proposed method on real workloads. As in the previous experiment, measurements were done with a fixed operating frequency (2.66GHz) and frequencies dynamically controlled with our proposed method. To experiment real-world workload fluctuation, we synthesized the transaction arrival ratio based on WorldCup98[2], the trace dataset of real Web servers during the World Cup in 1998.

We plotted the results in Fig. 6. "WO/ power saving" stands for the fixed frequency (2.66GHz), and "W/ AA power saving" for our application-aware power saving method with DVFS. As shown in Fig. 6(d), the processor ran at the minimum frequency for 66% of total execution time. When the arrival rate was about 15000tpmC, throughput differed little. When the arrival rate was about 35000tpmC, throughput degradation due to DVFS was about 5%. This degradation was caused by operating frequency flapping due to oscillations of transaction arrival rate. It is expected that lowering R_{down} would suppress this flapping but result in less power saving. Thresholds $R_{\text{up}}, R_{\text{down}}$ should be configured with consideration for the tradeoffs between performance and power saving requirements. As shown in Fig. 6(f), transaction response time was controlled based on the given provisioning thresholds so as to save power as much as possible.

Throughout the execution, power consumption was reduced by 11.6 watts on average, which accounted for 7.6% of total power consumption. As for the effect on application performance, throughput degradation was 4.1% on average.

6 Related Work

Power saving methodologies with architecture-level DVFS have been investigated under various constraints and with various objectives. Especially multicore DVFS control has been a mainstream field of research in recent years. In [8], Herbert et. al. examined the tradeoffs of granularity of the VFI(voltage/frequency island), which is a unit of voltage and frequency control. They also proposed in [7] that DVFS taking account of process variation of processor cores could improve energy efficiency. Prior to this research, evaluations of DVFS control



(a) arrival rate
(b) throughput
(c) response time
(d) operating frequency
(e) power consumption
(f) cumulative distribution of response time

Fig. 6. Experimental results with the real-world workload

algorithms had been done with benchmarks like SPEC2000, but in [7, 8], they were done with more realistic server workloads using Apache, TPC-C, TPC-H and so on.

As for power saving technologies in the field of database systems, The Claremont Report on Database Research 2005 [1], which has had a great impact on the future direction of database research, pointed out the importance of power saving in database systems. In [5], Graefe suggested that database systems should adopt energy efficiency as a metric for optimizations. He also presented challenges to be addressed regarding some components in database systems such as the query optimizer, the query scheduler, and so on. In [6], Harizopoulos et. al. pointed out that hardware-based power saving alone was not enough for power saving of database servers and that refinements of database systems were required.

Among database related research fields, online analytical processing (OLAP) is somewhat advanced in terms of power saving initiatives. Henkel et. al. measured the power consumption of each component in a typical server used for OLAP, and studied hardware configurations with optimal power consumption in [11]. Poess et. al. estimated power consumption of top systems with the TPC-H benchmark and did trend analysis of energy efficiency in [14]. They also examined power performance tradeoffs with various hardware configurations in [13]. Dmitris et. al. figured out the power consumption of basic operations in database systems such as hash join and sort-merge join, and proposed guidelines for energy-efficient system design in [17]. In [9], Lang et. al. proposed a power saving methodology using the power saving mode of processors and modifications in query schedulers. Xu et. al. proposed a power-aware query optimizer constructed by modeling power consumption of database servers [18].

As for power saving in OLTP, Poess et. al. estimated power consumption of top systems with the TPC-C benchmark, and pointed out the need for power saving in OLTP by predicting that the absolute amount of power consumption would continue to increase because the throughput-per-power increase rate could not compensate for the current performance increase rate [12]. Xu et. al. evaluated their power-aware query optimizer with TPC-C in [18], though their main target was decision support systems.

As for application-aware power saving approaches, Chen et. al. evaluated the effect of DVFS on performance in multitier applications [3] though they focused on applications like web services. In comparison with web services, OLTP is much more performance sensitive. Our paper proposes and verifies power saving mainly for such applications. Lee et. al. proposed power saving of video playing with DVFS [10]. Their idea is to introduce task scheduling into video players for reducing power consumption.

7 Conclusion

We proposed an application-aware power saving for OLTP, and evaluated it. By scaling system performance using application performance information, our proposed method enables a reduction in power consumption of performance intensive OLTP applications. As a result of our experimental evaluation with actual server workloads, 7.6% of total power consumption was reduced.

We are planning to extend our application-aware power-saving method and apply it not only to OLTP but to other I/O intensive applications such as web services so on.

References

1. Agrawal, R., Ailamaki, A., Bernstein, P.A., Brewer, E.A., Carey, M.J., Chaudhuri, S., Doan, A., Florescu, D., Franklin, M.J., Garcia-Molina, H., Gehrke, J., Gruen-

wald, L., Haas, L.M., Halevy, A.Y., Hellerstein, J.M., Ioannidis, Y.E., Korth, H.F., Kossmann, D., Madden, S., Magoulas, R., Ooi, B.C., O'Reilly, T., Ramakrishnan, R., Sarawagi, S., Stonebraker, M., Szalay, A.S., Weikum, G.: The claremont report on database research. SIGMOD Rec. 37(3), 9–19 (2008)

2. Arlitt, M., Jin, T.: Workload characterization of the 1998 world cup web site. Tech. rep., Hewlett Packard Laboratories Palo Alto (September 1999), http://tinyurl.com/23ftxrl
3. Chen, S., Joshi, K.R., Hiltunen, M.A., Schlichting, R.D., Sanders, W.H.: Blackbox prediction of the impact of dvfs on end-to-end performance of multitier systems. SIGMETRICS Perform. Eval. Rev. 37(4), 59–63 (2010)
4. EPA: Epa report to congress on server and data center energy efficiency. Tech. rep., U.S. Environmental Protection Agency (2007), http://tinyurl.com/2jz3ft
5. Graefe, G.: Database servers tailored to improve energy efficiency. In: Apel, S., Rosenmüller, M., Saake, G., Spinczyk, O. (eds.) Software Engineering for Tailor-made Data Management. pp. 24–28. University of Magdeburg (2008)
6. Harizopoulos, S., Shah, M.A., Meza, J., Ranganathan, P.: Energy efficiency: The new holy grail of data management systems research. In: CIDR 2009, Fourth Biennial Conference on Innovative Data Systems Research, Asilomar, CA, USA, January 4-7, 2009, Online Proceedings (2009)
7. Herbert, S., Marculescu, D.: Variation-aware dynamic voltage/frequency scaling. In: High Performance Computer Architecture, 2009. HPCA 2009. IEEE 15th International Symposium on. pp. 301 –312 (Feb 2009)
8. Herbert, S., Marculescu, D.: Analysis of dynamic voltage/frequency scaling in chip-multiprocessors. In: ISLPED '07: Proceedings of the 2007 international symposium on Low power electronics and design. pp. 38–43. ACM, New York, NY, USA (2007)
9. Lang, W., Patel, J.M.: Towards eco-friendly database management systems. In: CIDR 2009, Fourth Biennial Conference on Innovative Data Systems Research, Asilomar, CA, USA, January 4-7, 2009, Online Proceedings (2009)
10. Lee, W.Y., Ko, Y.W., Lee, H., Kim, H.: Energy-efficient scheduling of a real-time task on dvfs-enabled multi-cores. In: ICHIT '09: Proceedings of the 2009 International Conference on Hybrid Information Technology. pp. 273–277. ACM, New York, NY, USA (2009)
11. Meza, J., Shah, M.A., Ranganathan, P., Fitzner, M., Veazey, J.: Tracking the power in an enterprise decision support system. In: Henkel, J., Keshavarzi, A., Chang, N., Ghani, T. (eds.) ISLPED. pp. 261–266. ACM (2009)
12. Poess, M., Nambiar, R.O.: Energy cost, the key challenge of today's data centers: a power consumption analysis of tpc-c results. Proceedings of VLDB Endowment 1(2), 1229–1240 (2008)
13. Poess, M., Nambiar, R.O.: Tuning servers, storage and database for energy efficient data warehouses. In: Proceedings of the 26th International Conference on Data Engineering, ICDE 2010, March 1-6, 2010, Long Beach, California, USA. pp. 1006–1017. IEEE (2010)
14. Poess, M., Othayoth Nambiar, R.: A power consumption analysis of decision support systems. In: WOSP/SIPEW '10: Proceedings of the first joint WOSP/SIPEW international conference on Performance engineering. pp. 147–152. ACM, New York, NY, USA (2010)
15. Network Power, E.: Energy logic: Reducing data center energy consumption by creating savings that cascade across systems. White paper, Emerson Electric Co. (2009), http://tinyurl.com/7dhks3
16. Shanley, K.: Tpc releases new benchmark: Tpc-c. SIGMETRICS Performance Evaluation Review 20(2), 8–9 (1992)
17. Tsirogiannis, D., Harizopoulos, S., Shar, M.A.: Analyzing the energy efficiency of a database server. In: SIGMOD '10: Proceedings of the 36th SIGMOD international conference on Management of data. ACM, New York, NY, USA (2010)
18. Xu, Z., Tu, Y.C., Wang, X.: Exploring power-performance tradeoffs in database systems. In: Proceedings of the 26th International Conference on Data Engineering, ICDE 2010, March 1-6, 2010, Long Beach, California, USA. pp. 485–496. IEEE (2010)