` International Journal of Grid and Distributed Computing
Vol. 9, No. 9 (2016), pp.223-238
http://dx.doi.org/10.14257/ijgdc.2016.9.9.20

Using Frequency Scaling on Virtualized Memory in Cloud Datacenters

Yuan Tian[1, *] and Ze Xiao[2]

1School of Computer and Communication, Hunan Institute of Engineering, 411104, Hunan Province, China

2Department of Computer Technology and Application, Qinghai University, 810016, Qinghai Province, China
chenruntong89@126.com, xiaoze1992@gmail.com

Abstract

As the increasing of IT-infrastructure in cloud platforms, rapidly growth of energy consumption becomes a critical problem in many cloud datacenters. Conventionally, most of studies on energy-efficiency optimization concentrate on CPU related energy costs instead of memory subsystem, since CPU often dominates the total energy consumption in modern servers. However, such a situation is gradually changing as more and more cloud datacenters are equipped with larger and larger memory systems for dealing with data-intensive applications. In this paper, we present a novel mechanism, namely frequency scaling on virtualized memory (FSVM), which applies DVFS technology on memory subsystem based on the characteristics of active VM instances. Comparing with previous studies, our approach provides a fine-grained memory energy consumption conservation mechanism for virtualized servers. Extensive experiments are conducted to investigate the effectiveness and performance of our FSVM, and the results indicate that it can significantly improve the energy-efficiency of memory subsystem in virtualized servers.

Keywords*: cloud computing; virtualization; virtual machine; datacenter*

1. Introduction

Cloud computing has emerged as a promising high-performance computing platform, which provides a scalable and flexible computing platform for various kinds of applications [1]. In cloud computing environments, high-performance datacenters play a key role for resource management and service provision, and their performance and reliability directly affect the QoS (Quality of Ser-vice) satisfaction of cloud users [2]. To maintain desirable QoS, cloud providers tend to equip their datacenters with more advanced IT devices and keep them available for users in 24 hours [3, 4]. As a result, energy consumption of IT-infrastructures grows quickly, which significantly increases the operational costs of cloud providers [5].

Historically, most of studies focus on CPU related energy consumption since it has dominated energy consumption in servers. However, as processors have become more energy-efficient and more effective at managing their own power consumption, their contribution has been decreasing. In contrast, main memory energy consumption has been growing rapidly [5, 6], as multi-core servers are requiring increasing main memory bandwidth and capacity. In addition, memory energy management is challenging in the context of servers with modern DRAM technologies. According to the recent studies [6, 7], main memory in massive server platforms accounts for up to 40% of server energy which is comparable to or slightly higher than the energy consumption contribution of CPUs. In cloud datacenters, the fraction attributable to memory accesses may be even higher, because resource virtualization technology will introduce extra energy

ISSN: 2005-4262 IJGDC

consumption on the memory subsystem in servers [8-9].

In the past few years, memory energy conservation focused on creating memory idleness by scheduling, batching, and layout transformations, so that idle low-power states could be exploited [10-11]. Other works suggested reducing the number of DRAM chips that are accessed at a time [7, 12]. Generally speaking, the common ideology of these works is to reduce the number of chips or bits actually touched as a result of a memory access, thereby reducing the dynamic memory energy consumption. Unfortunately, the energy consumption conservation of those works becomes very weak in cloud platforms especially when virtualization platforms are involved. For example, data-intensive applications are very common in cloud platforms, which often lead to continuous and extensive I/O operations. As a result, creating enough idleness is difficult in memory subsystems in this case, since power management is available only at coarse granularity. So, deep idle low-power states can rarely be used without excessively degrading performance. In this work, we present a novel mechanism, namely frequency scaling on virtualized memory (FSVM), which applies DVFS technology on memory subsystem based on the characteristics of active VM instances. Comparing with previous studies, our approach provides a fine-grained memory energy consumption conservation mechanism for virtualized servers.

The rest of this paper is organized as follows: Section 2 presents the related work; Section 3 introduces the performance and energy consumption model in virtualized servers. Section 4 presents the memory energy optimization technology based on frequency scaling. In Section 5, extensive experiments are conducted to verify the effectiveness and performance of the proposed technology. Finally, Section 6 concludes the paper with a brief discussion of future work.

2. Related Work

In cloud computing environments, energy-efficiency optimization is a complex problem, since better tradeoffs between performance and energy consumption are often difficult to be obtained. So, various researchers take their efforts on different aspects. For example, many efforts have been taken into designing energy-aware scheduling algorithm for computation-intensive applications. For example, Bertran *et al.* proposed high accuracy VM power model [13]. Also they showed that Dynamic Voltage and Frequency Scaling (DVFS) mechanism will not affect the accuracy of the models. In [14], the authors suggested using traditional Max-Min algorithm with DVFS mechanism for scheduling workflow applications. In [15], the author presented analytical models based on an energy consumption metric to analyze the impact of dynamic frequency scaling on the energy consumption of various architectural design choices for hybrid-architecture chips. The power consumption implications of different processing schemes and various chip configurations were also analyzed. The analysis shows that by choosing the optimal hardware configuration, the energy savings can be increased considerably while keeping sacrifices in performance at tolerable levels. In [16], the authors proposed a scheduling algorithm for the cloud datacenters with a dynamic voltage frequency scaling technique. Its scheduling algorithm can efficiently increase resource utilization so as to decrease the energy consumption for executing jobs.

On the other side, plenty of efforts are also taken into reduce data-accessing energy consumed by storage or memory subsystems. For example, Manzanares *et al.* introduced a novel architecture called PRE-BUD, which provides significant energy savings for parallel I/O systems using buffer disks while maintaining high performance [17]. PRE-BUD can save energy by keeping data disks in the standby state for increased periods of time. In [18], the authors implemented an adaptive energy-saving scheme in parallel disk systems. Its key ideaology is that adaptability in energy conservation can be achieved through the integration of a dynamic disk scheduling scheme and power management in

parallel disk systems. In [19], the authors proposed a broadcast tree-based network for snoopy cache coherent multicores in memory shared systems. Those studies were effective for reducing the energy consumption of storage systems at hardware level. However, they were only suitable for coarse-grained energy management.

Recently, some researchers have taken their efforts on the relationship between energy consumption and features of applications. For example, Katsaros *et al*. studied the effects of user's QoS requirements on the server's energy consumption [20]. Based on extensive observations on several cloud platforms, they presented a service framework that allows monitoring the energy consumption of a cloud infrastructure, calculating its energy efficiency, and evaluating the gathered data in order to put in place an effective VM management. In [21], the authors analyzed behaviour of the memory power consumption when running different benchmark programs on multi-core servers. Their study showed that as memory data rate, capacity and bandwidth are being pushed higher and higher, the power consumption of memory systems becomes a significant part in the overall system power profile. Also, they proposed a mini-rank architecture for DDR3 memories to reduce memory power consumption by breaking each DRAM rank into multiple narrow mini-ranks and activating fewer devices for each request. Such a mini-rank design is also able to obtain better performance-power tradeoff for mixed workloads based on the memory access behavior and bandwidth requirement.

3. Energy and Performance Models in Virtualized Servers

3.1. Overview of Memory System

In modern servers, the memory subsystem is generally organized as a multi-level architecture, in which Memory Controller (MC) is responsible for sending commands to Dual In-line Memory Modules (DIMM) when events of last-level-cache missing occur in CPU. To improving the efficiency of memory accessing, the memory bus is split into multiple channels, which operate independently and can access disjoint regions of the physical address space in parallel. In each DIMM, there are multiple DRAM chips and a Phase-Lock-Loop device which can be used for maintaining frequency and phase synchronization. The DRAM chips are then organized as a set of ranks, each containing multiple two-dimensional memory arrays (often called banks).

Due to the increasing performance and capacity of memory subsystem, its energy consumption becomes more and more significantly comparing with that of CPU. So, it is important to understand the energy consumption distribution of memory subsystem. Typically, the DRAM energy consumption can be categorized into four classes: background, activation/precharge, read/write, and termination powers. The background power depends on the peripheral circuitry, transistor leakage, and refreshes operations. The activation/pre-charge power comes from the accessing operations on the memory arrays. The read/write power is due to these column accesses to row buffers. The termination power is due to terminating signals of other ranks on the same channel. It is clear that only the background power is independent of the memory accessing operations, so they are often called dynamic DRAM power. As shown in existing studies, the characteristics of up-level workloads have significant effects on the energy consumption of memory subsystem, especially for the part of dynamic DRAM power. For example, memory-intensive workloads will significantly increase the activation/pre-charge and read/write powers.

3.2. Performance Model of Virtualized Servers

In cloud datacenters, virtualization technology has been widely applied for providing securable and extendible services for up-level users. As a result, VM is used as the basic unit for constructing cloud applications. In a virtualized server, there exists a multi-

layered software stack as shown in Figure 1, which consists of physical devices, native OS, drivers, VM hypervisor, VM utilities and multiple VMs.

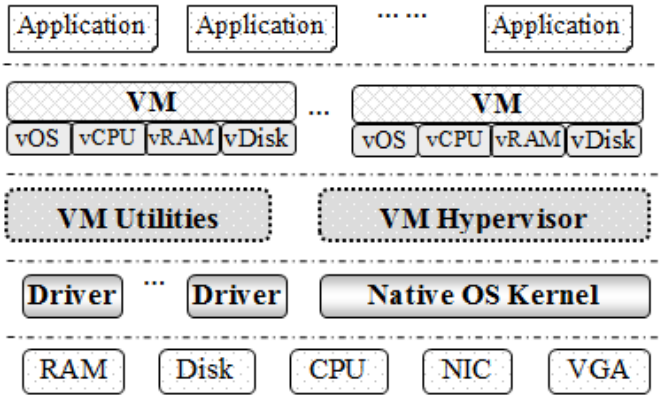


Figure 1. The Layered Framework of a Virtualized Server

In this framework, VM hypervisor and VM utilities can control over physical devices through driver modules or hardware interfaces directly. Because of the co-existing of multiple VM instances in the same physical, it is difficult to describe the performance model of each VM (user application) if not impossible. In this paper, our goal is to determine the runtime and power/energy implications of changing memory performance. So, the first step of doing that is to create a per-VM performance model in which the memory accessing operations can be correctly evaluated. Fortunately, modern CPUs have incorporated a set of Performance Monitor Counters (PMC) which can be used for analyzing the runtime characteristics of each VM instance. Using PMC facility for performance evaluation is not a new idea. However, most of those existing studies focused on CPU-related performance metrics. In our work, we mainly concentrate on the memory-related performance metrics. More importantly, our performance model is based on per-VM manner instead of the overall physical server.

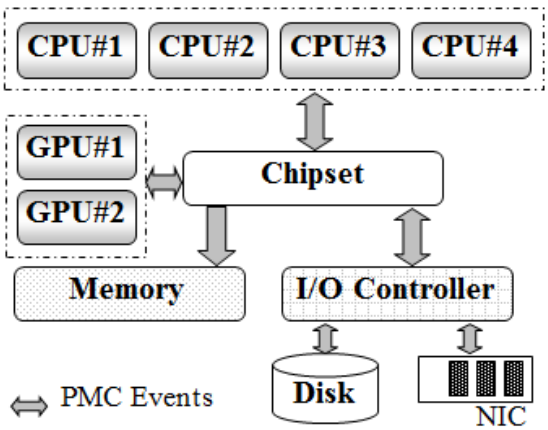


Figure 2. PMC Event Propagation in a Server

Typically, the PMCs and their respective events can be categorized into many classes according to their relationship to specific components, such as CPU, GPU, chipset, RAM, I/O controller, and disk. In the Figure 2, we demonstrate the propagation of PMC events between physical components. To preciously modeling the VM performance, we select the following counters: *Total Instructions Committed* (TIC), *Total LLC Misses* (TLM),

Bank Transactions Outstanding (BTO), Channel Transactions Outstanding (CTO), *Bank Transaction Counter* (BTC), *Channel Transactions Counter* (CTC), *Row Buffer Hit Counter* (RBHC), *Open Row Buffer Miss Counter* (OBMC), *Closed Row Buffer Miss Counter* (CBMC), *Exit PowerDown Counter* (EPDC), *Precharge Time Counter* (PTC), *Page Open/Close Counter* (POCC). The detailed description of these PMCs can be found in [22].

At first, the total execution time of a VM instance can be described as

$$T_i^{exec} = c_i^{cpu} \cdot \tilde{t}_{cpu} + c_i^{mem} \cdot \tilde{t}_{mem} \tag{1}$$

where c_i^{cpu} is the number of instructions executed by processors, c_i^{mem} is the number of TLM events, $\tilde{t}_{cpu}$ is the average time that instruction spend on CPU, and $\tilde{t}_{mem}$ is the average time that a TLM instruction spends on MC and CPU. By the above mentioned PMCs, c_i^{cpu} and c_i^{mem} can be easily obtained through accumulating the corresponding counters. As to $\tilde{t}_{cpu}$, it also can be obtained by execution a set of benchmarks.

So, the key challenging of this per-VM model is how to evaluate $\tilde{t}_{mem}$ because time of TLM instructions varies with memory subsystem frequency. As mentioned in Section 3.1, the memory accessing operation can be divided into two phases: waiting bus and data transferring. So, we can use the expected times of waiting bus and data transferring to describe it, that is

$$\tilde{t}_{mem} = E[T_{bus}] + E[T_{data}] \tag{2}$$

In the $E[T_{data}]$, it consists of the time of accessing a particular bank (noted as T_{bank}) and the waiting time of MC (noted as T_{MC}). T_{bank} can be considered as a function of memory's parameters and VM's runtime characteristics. By referring the PMCs and benchmark profiling, we can obtain the expected T_{bank} as following:

$$E[T_{bank}] = \frac{\sum_{i \in S^{bank}} c_i \cdot t_i}{\sum_{i \in S^{bank}} c_i} \tag{3}$$

where S^{bank} ={BTO, RBHC, OBMC, EPDC} is the set of PMCs that involve in the banking accessing operations, c_i is the event number of each PMC, t_i is time of the PMC event. As to T_{MC}, it varies as a function of MC frequency that we will discussed in Section 4. By the same way, we can describe the $E[T_{bus}]$ as

$$E[T_{bank}] = \frac{\sum_{i \in S^{bus}} c_i \cdot t_i}{\sum_{i \in S^{bus}} c_i} \tag{3}$$

where S^{bus} ={BTO, RBHC, CTO, CBMC, PTC, POCC}.

3.3. Energy Model of Memory

Conventionally, energy model of a server can be modeled as the function of its component's utilization. However, such a utilization-based energy model will lead to significant error when resource virtualized technology is used. In this work, we still use PMCs for describing the energy model of a virtualized server. Since only part of PMCs is suitable for describing the energy consumption of a server, those rPMCs are carefully picked out by our experiments on different benchmarks. The selected PMCs are uOps,

TLM, TLB, Halt, FSB, DMA, and Interrupt. To determine the best operating point of memory subsystem, we introduce a novel energy-efficiency measurement, called virtual Memory Energy-consumption Ratio (vMER), which uses the working frequency of MC to predict the energy usage of memory subsystem.

$$vMER = \frac{\sum_i \left(T_i^{exec} \cdot P_i^{mem} \right)}{\sum_i \left(T_i^{exec} \cdot P_i^{vm} \right)} \tag{4}$$

where P_i^{mem} is the memory power model of a given VM instance, P_i^{vm} is the total power model of this VM instance. In a virtualized server, the P_i^{mem} and P_i^{vm} can be modeled as following:

$$P_i^{mem} = \sum_{i \in \{TLM, TLB, FSB, DMA\}} (k_i \times c_i) \tag{5}$$

$$P_i^{vm} = P_i^{mem} + P_i^{cpu} + P_i^{disk} + P_i^{io} \tag{6}$$

where k_i is empirical coefficient for per-VM power model which can be obtained by profiling benchmarks, c_i is the event number of related PMC. In equation (6), we only use PMC-based power model for memory subsystems. As to P_i^{cpu}, P_i^{disk} and P_i^{io}, we still use typical utilization-based power models as shown in [23].

4. Memory Energy-Efficiency Optimization Policy

4.1. DVFS on Memory Subsystem

In modern servers, DVFS mechanism has been incorporated in both CPU and memory components. Although DVFS mechanism for CPU energy conservation has been extensively studied in the past decades, using it on memory energy conservation is quite seldom. Generally speaking, DVFS mechanism on memory enables us to adjust the working frequency of MC, memory bus, DIMM, and DRAM chips. For example, lowering the frequencies of those components will resulting longer data bursts and MC delay, which in turn increases the memory accessing time. As a result, the power consumption of memory subsystem can be reduced at the costs of performance degradation.

However, according to the recent study, accessing latency in the stages of memory operations shows only minor increases in average memory access time, instead of linear increasing in overall memory access time. Therefore, using DVFS for memory power conservation can be a cost-effective approach. Here, we firstly summarize the ways that how to use DVFS for memory power conservation:

(1) Lowering down the frequency can increase read/write operation energy in a linear manner.

(2) The power consumption of MC component can be reduced by a cubic factor when lowering down the working frequency.

(3) The background and register/PLL power will be reduced linearly when lowering down the working frequency.

(4) If using DVFS on CPU for power conservation and not using it on memory subsystem, the degradation of application performance will be more cost-expensive.

Based on the above observations, in this work we present a novel mechanism, namely frequency scaling on virtualized memory (FSVM), which applies DVFS technology on memory subsystem based on the characteristics of active VM instances..

4.2. Share-Reclaiming Scheduling Policy

Currently, commercially-available DIMMs have already supported multiple working frequencies. However, the operation of switching frequency typically needs to reboot the target server. Therefore, it is difficult to apply DVFS mechanism directly on the DIMMs especially when the virtualization technology is involved. Fortunately, the supply voltage of MC component can be adjusted during the runtime, which provides us an alternative to change the working frequency of memory bus. So, the key idea of our approach is to adjust the supply voltage of the MC component with aiming at matching the virtual memory accessing with the working frequency of memory bus. According to the standard of DDR, the operating frequency of MC can be reset while in the pre-charge, power-down or self-refresh mode. So, we can configure the MC component at a new power working mode by shortly suspending the memory accessing operations. In such a mechanism, we need to use Phase-Lock Loop (PLL) and Delay-Lock Loop (DLL) devices for synchronizing signal frequency and phase between different memory components. So, the latency overheads of this mechanism are due to the synchronization time of PLL and DLL. For instance, the synchronization time of PLL or DLL is about 200~400 bus cycles.

Combining the mechanisms at both memory-level and CPU-level, our FSVM can be summarized into four phases: performance profiling, DVS on MC, bus frequency adjusting, and accessing time-slot configuring. In the phase of performance profiling, the target sever is profiled by collecting statistics using the PMCs. By using the performance and energy models described in Section 3.2 and Section 3.3, we can obtain the basic information of memory subsystem when it works at a given mode. In the phase of DVS on MC, we adjust the working voltage of MC component if the profiling information indicates that doing it can conserve the energy consumption. In the phase of bus frequency adjusting, the operations of memory accessing are temporarily halted and PLLs and DLLs are resynchronized. Finally, we update the accessing time-slot configuration which is execution time assigned to the VM instances.

In our FSVM mechanism, only the time-slot configuring phase is accomplished at resource virtualization level. Generally speaking, changing the accessing time-slot of VM instances will significantly affect the execution performance and energy consumption of applications. In order to reduce the performance degradation, we define a fixed threshold which indicates the execution time latency when using FSVM comparing with the case without using it. It is noteworthy that our FSVM queries the performance counters both at the end of each epoch and at the end of each profiling phase. As to the frequency selection of data bus, we always select a frequency that maximizes full-system energy savings. The energy-minimal frequency is not necessarily the lowest frequency as the system continues to consume energy when the performance of memory subsystem is reduced and lowering frequency can result in energy loss if the program slowdown is too high. As noted in Section 3, our models explicitly account for the balance between memory energy and application execution time.

5. Experiments and Performance Comparison

5.1. Experimental Settings

To examine the effectiveness of our FSVM policy, we conducted extensive experiments on various virtualized servers with different workloads. The virtualization platform used in our experiments is Xen-v4.1.2. In Table 1, we list the workloads used in the experiments and their characteristics.

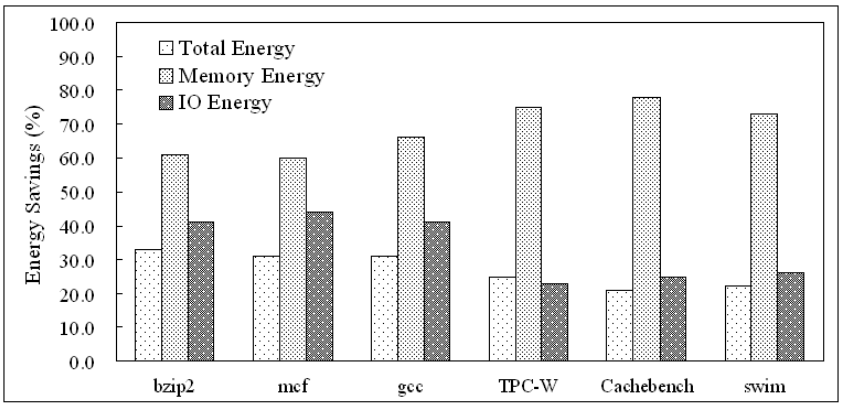
Table 1. Workloads and their Characteristics

Benchmark	Characteristics
bzip2	CPU and RAM intensive
mcf	CPU and RAM intensive
gcc	Balanced CPU and RAM, IO intensive
TPC-W	Balanced CPU and RAM, IO intensive
Cachebench	CPU moderate, RAM intensive
Swim	CPU moderate, RAM intensive

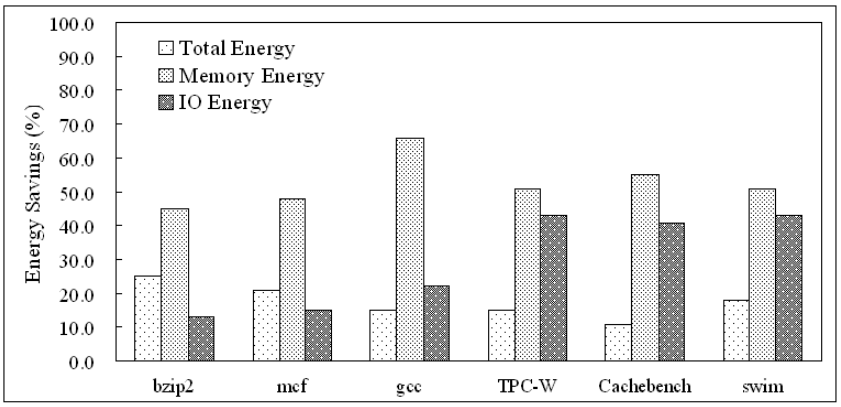
To collect the experimental results, we implement the FSVM policy as a driver model, in which we can take advantages of all the resources in the tested platform. For example, we can use Oprofile to sample PMC events and categorize the original reports produced on per-VM basis. Also, we simulate two scenarios for energy-efficiency comparison: (1) memory controller immediately transitions a rank to fast-exit power-down upon closing all banks (2) a fixed frequency for the entire memory subsystem is selected statically.

5.2. Evaluation on Energy and Performance

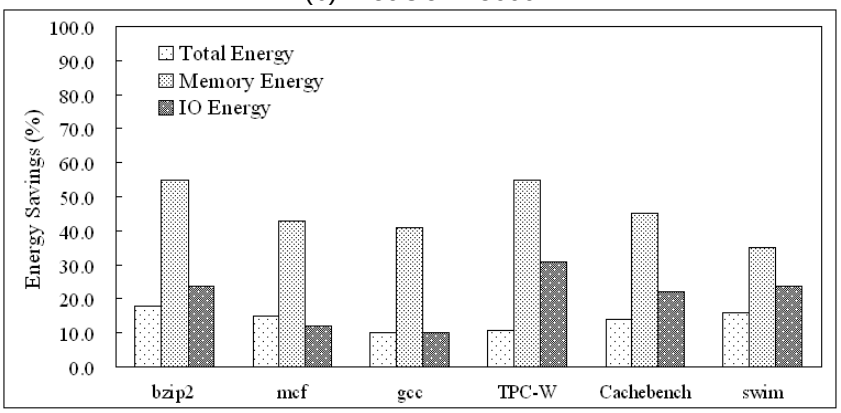
In the first set of experiments, we evaluate the impact of FSVM policy on the energy and performance. For the convenience of comparison, we set that the maximum allowable performance degradation is less than 15%, and baseline results is obtained by using Decoupled-DIMM approach [24] which is widely used for memory energy consumption conserving. All the experiments are conducted on four servers from different vendors (ThinkStation D20, Precision T5600, XASUN EX6, IdeaCentre K415), in which ThinkStation D20 and XASUN EX6 are high-performance workstations, and DELL Precision T6500 and IdeaCentre K415 are desktop PCs. The results are shown in Figure 3 and Figure 4.



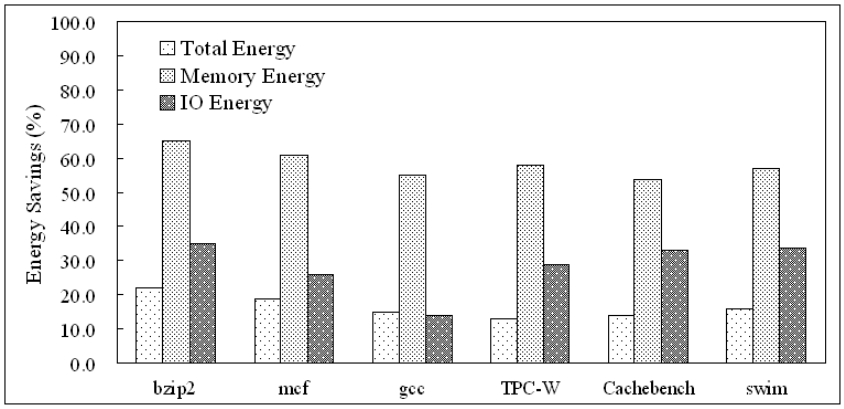
(a) ThinkStation D20



(b) Precision T5600



(c) XASUN EX6

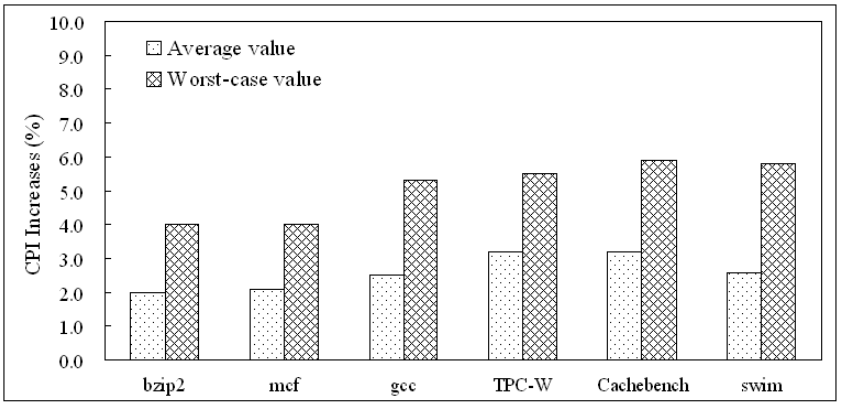


(d) IdeaCentre K415

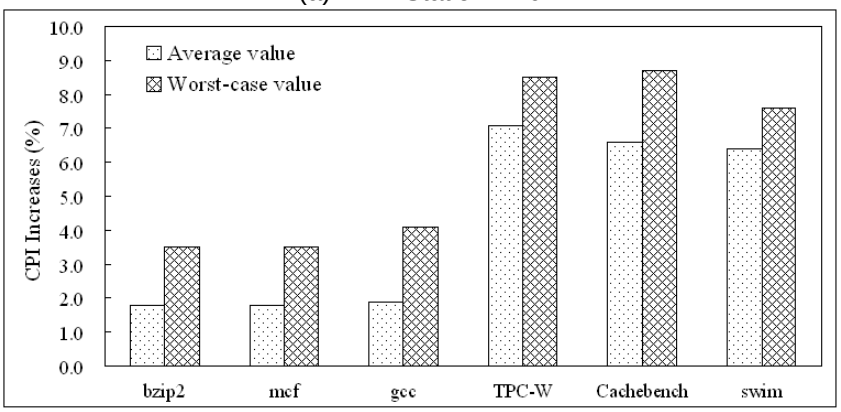
Figure 3. Energy Savings on Different Servers with Different Workloads

In Figure 3, we demonstrate the energy savings we achieve for each workload, compared to a baseline system that keeps the memory subsystem at its highest voltage and frequency. For clear representation, we separate the memory and IO energy from the total energy measurement. In all cases, we can see that the memory energy savings range from 31% to 73%, and the total energy savings range from 11% to 33%. When tested in ThinkStation D20, we noticed that the highest memory energy saving is obtained when using TPC-W and Cachebench workloads (above 70%). This is because that the workloads can keep the voltage and frequency of the memory system at their lowest possible value all the time. In addition, ThinkStation D20 is equipped with biggest memory and cache, which enables our FSVM policy to maximize its memory energy

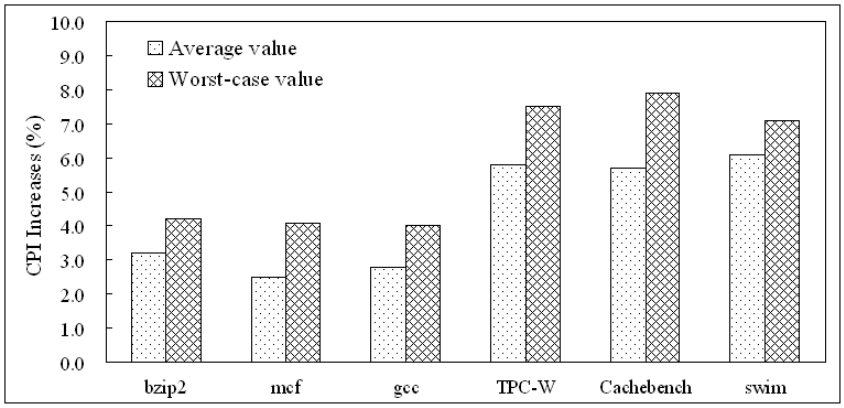
saving effects. The energy savings achieved on XASUN EX6 are generally lower that other tested servers. By carefully examine the logs of the experiments, we find that the memory channel traffic is more intensive than other servers even using the same workloads for benchmarking, which in turn reduces the opportunities for significant voltage and frequency scaling. One interesting finding in this set of experiments is that our FSVM policy can reduce the energy consumption of IO subsystems (including IO controller and hard disk), ranging from 11% to 40%.



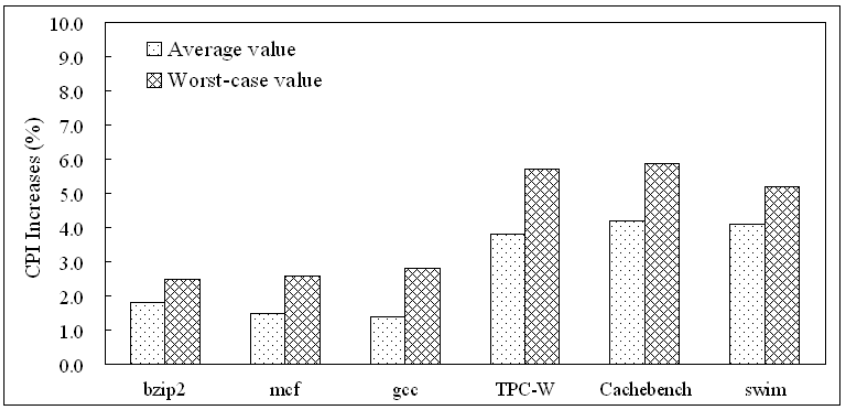
(a) ThinkStation D20



(b) Precision T5600



(c) XASUN EX6



(d) IdeaCentre K415

Figure 4. CPI Increases of FSVM on Different Servers with Different Workloads

To investigate the performance degradation of using FSVM, we use the CPI (CPU cycles Per Instruction) measurement to evaluate the overhead of the FSVM. In Figure 4, we show the energy savings can be achieved without violating the maximum allowable performance degradation (<10%) for all the tested workloads in various servers. The results show the average and maximum percent CPI losses for all the tested workloads comparing with the baseline approach. The results demonstrate that the FSVM is effective to limit the maximum CPI increase to the acceptable range in all cases. For example, the CPI measurements of all cases are never more than 8.7%. The results also demonstrate that, when we average the performance degradations of all the applications in each workload, this average is never higher than 7.0%. Among all the tested servers, we can see that IdeaCentre K415 performs best on CPI increases measurement, while Precision T5600 performs worst. With respect to different workloads, we noticed that CPU-intensive workloads generally perform better than those memory-intensive ones. These experimental results indicate that our FSVM can keep voltage and frequency low longer and approximate the maximum allowable degradation more closely. Thus, our policy degrades performance only up to the point that this translates into overall system energy savings.

5.3. Evaluation on Various Parameters

In this experiment, we investigate the effect of four parameters on the performance of our FSVM, including MAPD (maximum allowable performance degradation), DCN (DIMM channel numbers), MPM (configuration of memory power model), and PPM (power proportionality of MC). All the experiments are conducted on ThinkStation D20 server, and the tested workload consists of multiple benchmarks noted in Table 1 by mixing them on a single VM instance.

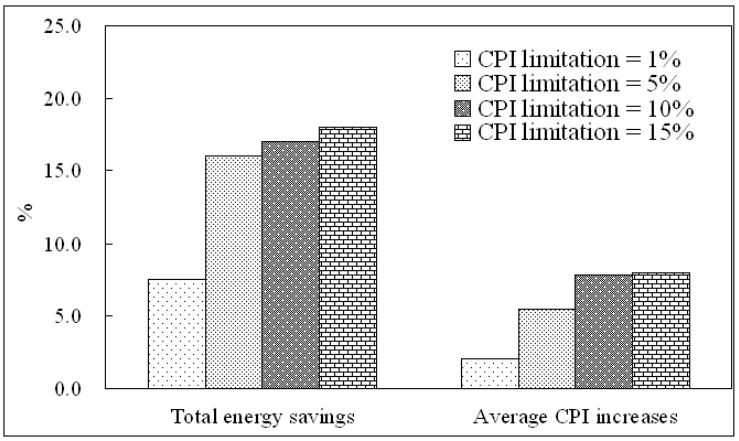


Figure 5. Effects of CPI Limitation on Energy Saving and Performance Degradation

MAPD parameter plays an important role in FSVM because higher allowable degradations could enable greater energy savings. To understand the impact of this parameter, Figure 5 illustrates the energy savings and maximum achieved degradations, where MAPD is gradually increased from 1% to 15%. As shown in Figure 5, we can see that 1% and 5% degradations indeed produce lower energy savings. However, allowing 10% or more degradation does not improve energy savings as we expected. This result indicates that prolonging the execution to conserve more memory energy actually increases overall energy if the MAPD parameter is above a certain value (the value is 10% in our experiment).

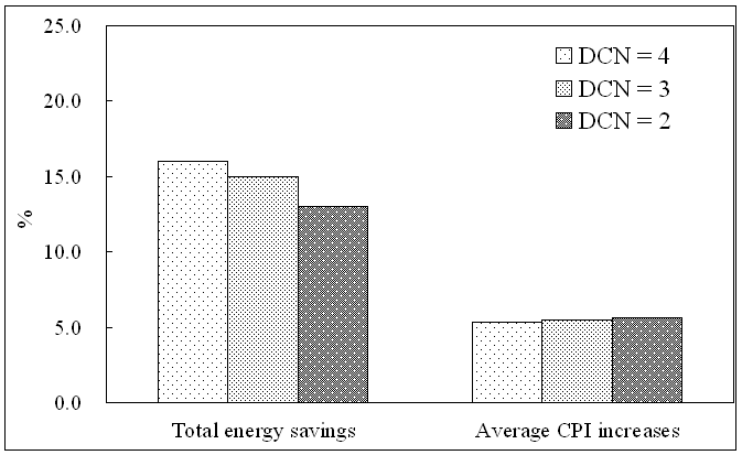


Figure 6. Effects of DIMM Channel Numbers on Energy Saving and Performance Degradation

When using FSVM policy, DCN parameter is the most important factor of the memory subsystem configuration. The number of channels directly affects how heavily utilized each channel is and, thus, our opportunities to lower frequency without excessively degrading performance. In fact, decreasing the number of channels approximates the effect of greater memory traffic that could result from prefetching or execution. In Figure 6, we depict the energy savings and maximum achieved performance degradation for different DCN value. The results show that increases in the number of channels indeed increase the benefits of FSVM by non-trivial amounts, without affecting our ability to limit performance losses. In addition, it also shows that doubling the channel traffic (from 4 to 2 channels) still leads to system energy savings of roughly 14%. Another approach

for studying the effect of greater memory traffic is to increase the number of cores, while keeping the LLC size the same. Thus, we performed experiments with 32 cores and 4 memory channels. For the mixed workloads, the larger number of cores causes more increases in traffic. These increases translate into system energy savings ranging from 7.1% to 11.4%, without any violations of the performance bound.

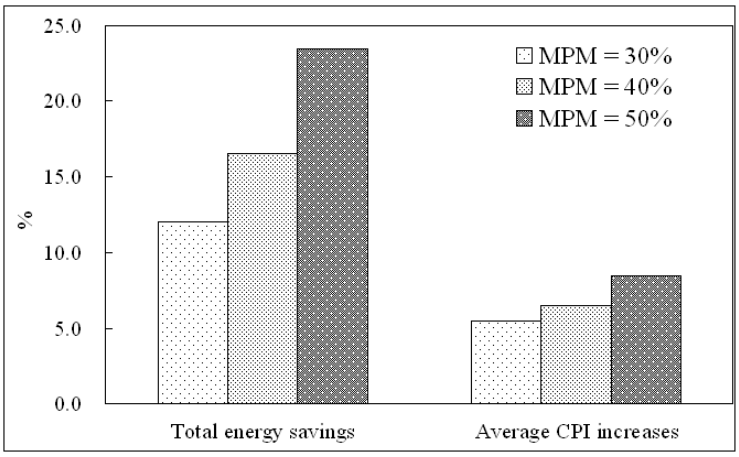


Figure 7. Effects of Memory Power Model on Energy Saving and Performance Degradation

As the FSVM is designed to reduce the memory subsystem energy, the contribution of the memory subsystem to the overall power consumption of the server becomes a crucial parameter. Intuitively, the larger MPM value will result in more energy savings. In Figure 7, we depict the impacts of different MPM value (ranging from 30% to 50%) on the system energy savings and their performance degradations. Recall that our baseline assumes a fraction of 40%. The figure shows that the fraction of memory power has a significant effect on the system energy savings. By increasing the fraction from 30% to 50%, we can obtain significant improvement on energy savings (from 12% vs 23.4%). The maximum CPI degradation increases by a few percent as well, but stays within the allowed range.

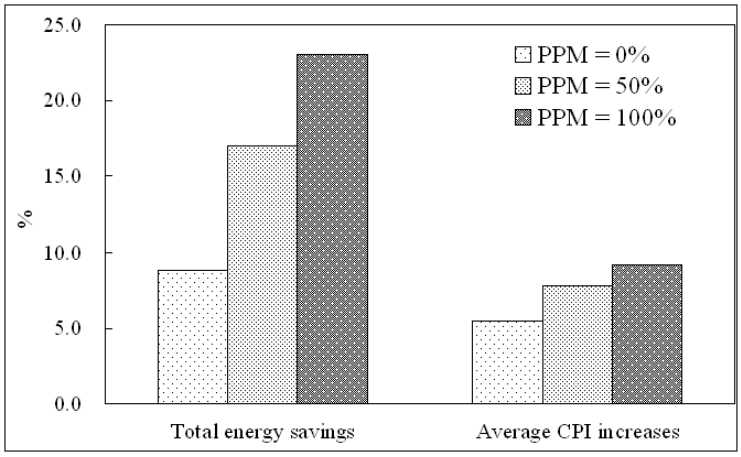


Figure 8. Effects of Power Proportionality of MC on Energy Saving and Performance Degradation

As the memory controller designs are vendor-dependent, we studied different value of PPM for these components. Specifically, we varied their idle power consumption from

0% to 100% of their peak power consumption. For the convenience of comparison, we assume that their power consumption changes linearly with utilization between idle and peak loads. In Figure 8, we depicts the effects of power proportionality of MC on energy saving and performance degradation. The result shows that the power proportionality of these components has a significant impact on the system energy savings. Interestingly, decreasing proportionality actually increases our savings significantly to 22.5%. As the memory subsystem's idle power increases (decreasing proportionality), the scope of FSVM to reduce register and memory controller power grows accordingly. Importantly, FSVM achieves these benefits without violating the allowed performance degradation.

6. Conclusion

In this work, we proposed a frequency scaling on virtualized memory (FSVM) policy, which applies DVFS technology on memory subsystem based on the characteristics of active VM instances. Comparing with previous studies, our approach provides a fine-grained memory energy consumption conservation mechanism for virtualized servers. Extensive experiments are conducted to investigate the effectiveness and performance of our FSVM, and the results indicate that it can significantly improve the energy-efficiency of memory subsystem in virtualized servers. Currently, the FSVM is implemented as a driver model in host operation system on virtualized server. In the future, we plan to re-design it as a plug-in model in some virtualized platforms (*i.e.* Xen or VMware). Also, we are planning to investigate the energy conserving effects when using different VM scheduler. By doing this, we are expecting to improve the current VM scheduling algorithm and provide some memory energy-aware VM scheduler.

References

[1] Williams, D., Jamjoom, H. "Plug into the Supercloud." IEEE Internet Computing, vol.17, no.2, pp.28-34, 2012.
[2] Wang, W.-J., Chang, Y.-S. "Adaptive scheduling for parallel tasks with QoS satisfaction for hybrid cloud environments." Journal of Supercomputing, vol.66, no.2, pp.783-811, 2013.
[3] Pedersen, J.M., Riaz, M.T. "Using latency as a QoS indicator for global cloud computing services." Concurrency and Computation: Practice & Experience, vol.25, no.18, pp.2488-2500, 2013.
[4] Zheng, Z., Wu, X. Zhang, Y., et al. "QoS Ranking Prediction for Cloud Services." IEEE Transactions on Parallel and Distributed Systems, vol.24, no.6, pp.1213-1222, 2013.
[5] Mach, W., Schikuta, E. "Toward an economic and energy-aware cloud cost model." Concurrency and Computation: Practice & Experience, vol.25, no.18, pp.2471-2487, 2013.
[6] Long, S., Zhao, Y., Chen, W. "A three-phase energy-saving strategy for cloud storage systems." Journal of Systems and Software, vol.87, no.2, pp.38-47, 2014.
[7] Gaona, E., Titos, R., Fernandez, J. "On the design of energy-efficient hardware transactional memorysystems." Concurrency and Computation: Practice & Experience, vol.25, no.6, pp.862-880, 2013.
[8] Lovasz, G., Niedermeier, F. "Performance tradeoffs of energy-aware virtual machine consolidation." Cluster Computing, vol.16, no.3, pp.481-496, 2013.
[9] Xiao, P., Hu, Z.G. Zhang, Y.P. "An energy-aware heuristic scheduling for data-intensive workflows in virtualized datacenters." Journal of Computer Science and Technology, vol.28, no.6, pp.948-961, 2013.
[10] Tolentino, M.E., Turner, J., Cameron, K.W. "Memory MISER: improving main memory energy efficiency in servers." IEEE Transactions on Computers, vol.58, no.3, pp.336-350, 2009.
[11] Ahn, J.H., Jouppi, N.P., Kozyrakis, C., et al. "Improving system energy efficiency with memory rank subsetting." ACM Transactions on Architecture and Code Optimization, vol.9, no.1, pp.1-35, 2012.
[12] Gaona, E., Titos-Gil, J.R., Fernandez, J., et al. "Selective dynamic serialization for reducing energy consumption in hardware transactional memory systems." Journal of Supercomputing, vol.68, no.2, pp.914-934, 2014.
[13] Bertran, R., Becerra, Y., Carrera, D., et al. "Energy accounting for shared virtualized environments under DVFS using PMC-based power models." Future Generation Computer Systems, vol.28, no.2, pp.457-468, 2012.
[14] Rizvandi, N.B., Taheri, J., Zomaya, A.Y. "Some observations on optimal frequency selection in DVFS-based energy consumption minimization." Journal of Parallel and Distributed Computing, vol.71, no.8, pp.1154-1164, 2011.
[15] Marowka, A. "Maximizing energy saving of dual-architecture processors using DVFS." Journal of

Supercomputing, vol.68, no.3, pp.1163-1183, 2014.
[16] Wu, C.M., Chang, R.S. "A green energy-efficient scheduling algorithm using the DVFS technique for cloud datacenters." Future Generation Computer Systems, vol.37, no.2, pp.141-147, 2014.
[17] Manzanares, A., Qin, X. "PRE-BUD: prefetching for energy-efficient parallel i/o systems with buffer disks." ACM Transactions on Storage, vol.7, no.1, pp.1-42, 2011.
[18] Nijim, M., Qin, X. "An adaptive energy-conserving strategy for parallel disk systems." Future Generation Computer Systems, vol.29, no.1, pp.196-207, 2013.
[19] Morris, R., Jolley, E., Kodi, A.K. "Extending the performance and energy-efficiency of Shared memory multicores with nanophotonic technology." IEEE Transactions on Parallel and Distributed Systems, vol.25, no.1, pp.83-92, 2014.
[20] Katsaros, G., Subirats, J. "A service framework for energy-aware monitoring and VM management in Clouds." Future Generation Computer Systems, vol.29, no.8, pp.2077-2091, 2013.
[21] Fang, K., Zheng, H., Lin, J., et al. "Mini-rank: a power-efficient DDRx DRAM memory architecture." IEEE Transactions on Computers, vol.63, no.6, pp. 1500-1512, 2014.
[22] Bertran, R., Becerra, Y., Carrera, D., et al. "Energy accounting for shared virtualized environments under DVFS using PMC-based power models." Future Generation Computer Systems, vol.28, no.2, pp.457-468, 2012.
[23] Davis, J.D., Rivoire, S. "Including variability in large-scale cluster power models." IEEE Computer Architecture Letters, vol.11, no.2, pp.29-32, 2012.
[24] Zheng, H., Lin, J., Zhang, Z., et al. "Decoupled dimm: building high-bandwidth memory system using low-speed DRAM devices." Proceedings of International Symposium on Computer Architecture, pp.35-44, 2009.

Authors

Yuan Tian, received her master degree in Xiangtan University. Currently, he works in Hunan Institute of Engineering as a senior lecturer. Her research interests include complex networking deployment, distributed computing, energy-efficiency optimization.

Ze Xiao, is currently studying in Qinhai University majoring in Computer Science and Technology. His research interests include distributed systems and cloud computing.