Software Thermal Management of DRAM Memory for Multicore Systems

Jiang Lin[1], Hongzhong Zheng[2], Zhichun Zhu[2], Eugene Gorbatov[3], Howard David[3] and Zhao Zhang[1]

[1]Department of Electrical and Computer Engineering
Iowa State University
Ames, IA 50011
{linj,zzhang}@iastate.edu

[2]Department of Electrical and Computer Engineering
University of Illinois at Chicago
Chicago, IL 60607
{hzheng2,zzhu}@uic.edu

[3]Corporation Technology Group
Intel Corp.
Hillsboro, OR 97124

{eugene.gorbatov,howard.david}@intel.com

ABSTRACT

Thermal management of DRAM memory has become a critical issue for server systems. We have done, to our best knowledge, the first study of software thermal management for memory subsystem on real machines. Two recently proposed DTM (Dynamic Thermal Management) policies have been improved and implemented in Linux OS and evaluated on two multicore servers, a Dell PowerEdge 1950 server and a customized Intel SR1500AL server testbed. The experimental results first confirm that a system-level memory DTM policy may significantly improve system performance and power efficiency, compared with existing memory bandwidth throttling scheme. A policy called DTM-ACG (Adaptive Core Gating) shows performance improvement comparable to that reported previously. The average performance improvements are 13.3% and 7.2% on the PowerEdge 1950 and the SR1500AL (vs. 16.3% from the previous simulation-based study), respectively. We also have surprising findings that reveal the weakness of the previous study: the CPU heat dissipation and its impact on DRAM memories, which were ignored, are significant factors. We have observed that the second policy, called DTM-CDVFS (Coordinated Dynamic Voltage and Frequency Scaling), has much better performance than previously reported for this reason. The average improvements are 10.8% and 15.3% on the two machines (vs. 3.4% from the previous study), respectively. It also significantly reduces the processor power by 15.5% and energy by 22.7% on average.

Categories and Subject Descriptors: B.3.2 [Primary Memory]: Design Styles

General Terms: Design, Management, Performance

Keywords: DRAM Memories, Thermal Management

SIGMETRICS'08, June 2–6, 2008, Annapolis, Maryland, USA.

1. INTRODUCTION

Thermal management has been a first-order consideration in processor and hard disk design for a long time, and now it has become critically important in the design of DRAM memory subsystems. This trend is driven by the wide adoption of multi-core processors and the ever increasing demand for high capacity, high bandwidth from memory subsystems. For example, a current small-scale, two-way SMP server [8] provides peak memory bandwidth of 24 GB/s and maximum memory capacity of 32 GB to support up to eight processor cores. The design of high bandwidth and high density DRAM subsystem leads to increasing power consumption and heat generation. The maximum power consumption of the memory subsystem can reach 100 watts, which is in the same range of the power consumed by the processors. With this degree of power consumption, DRAM power and thermal management has become an urgent and critical issue. While studies have done on DRAM power management, there lacks systematic research on DRAM thermal management.

Modern computing systems are designed with cooling capabilities to allow full system performance under normal operating conditions as well as thermal solutions to safeguard the systems against adverse situations. As for DRAM memory subsystems, a simple thermal solution has been used in servers [16]: when the memory subsystem is approaching a critical thermal threshold, the memory controller throttles the memory bandwidth. Another solution, proposed for mobile systems, shuts down a whole memory subsystem [12]. In those systems, memory thermal management is used as a protection mechanism that ensures safe operation and prevents thermal emergencies under abnormal scenarios. These scenarios while not common do occur in practice. They can be due to a poorly designed thermal solution in other subsystems, system fan failure, obstructions to airflow within a system, thermally challenging workload mix or other reasons that cause a system to operate outside of its thermal design boundaries. Thermal management is also necessary since time constants with which silicon components can cross a critical thermal limit are much faster than the response time of system-level cooling control such as fan.

Our research goal is to design and implement memory DTM methods for server systems running in *constrained thermal environments* to improve the systems' performance and/or power effi-

ciency. Thermal management can be a normal form when users or system operators make a decision to operate in more constrained thermal environments, including unavoidable high ambient temperatures, the need to limit fan speed for acoustic reasons, or the necessity to reduce cooling costs in data centers.

Throttling memory bandwidth under a certain threshold can prevent memory temperature from increasing, thus avoiding any possibility of thermal emergency. Nevertheless, it may limit the system performance unnecessarily. A carefully designed DTM (dynamic thermal management) scheme, when used in combination with bandwidth throttling, may improve system performance and/or improve system power efficiency without putting the system in hazard. A recent study [15] proposed two schemes that directly throttle the processor execution through core gating (selectively shutting down processor cores) or DVFS (dynamic voltage and frequency scaling) for memory thermal management. It evaluated those schemes using a dynamic DRAM thermal model and simulation, and reported that integrating processor power management and execution control with memory thermal management yields significant gains in system performance and energy efficiency. However, the work has two limitations. First, the DRAM thermal model used for evaluation of different thermal management mechanisms has not been validated on real systems. Given the dependency between the accuracy of the thermal model and reported power and performance gains, it is necessary to confirm results presented in the study by implementing and evaluating these approaches in real systems. Second, due to inherent limitations of running long workload traces in a simulator, the design space and parameters of the proposed schemes were not full explored and adequately analyzed.

To address these issues, we evaluate the existing memory DTM schemes on two server systems. Our study uses measurements on real systems running multiprogramming workloads. We have implemented these schemes in a Linux OS and evaluated their performance and power efficiency on two servers configured with latest generation hardware, a Dell PowerEdge 1950 server and an Intel SR1500AL server testbed (called PE1950 and SR1500AL thereafter). To obtain an accurate picture of the power and performance benefits of mechanisms evaluated in this paper, we instrumented the SR1500AL with power and thermal sensors to get fine-grain measurements at a component level.

To the best of our knowledge, *this is the first study of software thermal management for memory subsystem on real machines*. We have done comprehensive experiments and detailed analysis regarding performance, memory temperature profiles, and system power and energy consumption. Our experiments first confirm that the two recently proposed schemes significantly improve performance of real server systems in constrained thermal environments. In addition, we have encouraging findings that address the limitations of the previous work.

- Compared with the simple DTM-BW (DTM through memory Bandwidth Throttling) method, the DTM-ACG (DTM through Adaptive Core Gating) scheme [15] improves performance by up to 24.2% and 21.7% on the PE1950 and SR1500AL servers, respectively; and 13.3% and 7.2% on average, respectively. The improvements of DTM-CDVFS (DTM through Coordinated Dynamic Voltage and Frequency Scaling) are up to 18.0% and 23.9%, and 10.8% and 15.3% on average on the two servers, respectively.

- The performance gain of the DTM-CDVFS scheme measured on real systems is much better than the previously reported simulation result, which is only 3.4% on average. Besides the expected performance difference due to different configurations of the real systems and the simulated one, our analysis indicates that the CPU heat dissipation and its influence on DRAM, which were ignored in the previous study, is a significant factor.

- We have also found that the DTM-CDVFS method improves the system power efficiency in addition to the performance gains. It reduces the processor power rate by 15.5% on the SR1500AL. The energy consumed by the processor and memory is reduced by 22.7% on average.

- We have evaluated a new scheme, called DTM-COMB, that combines DTM-ACG and DTM-CDVFS. It may stop a subset of cores and apply DVFS to the others. Our experimental results show that the new scheme may further improve the performance by up to 5.7%.

When compared with the simulation-based work [15], this study is novel in memory DTM implementations and the experimental methods. It is the first effort showing that memory DTM can be implemented as part of OS power management and work in conjunction with existing power management mechanisms (e.g. DVFS). The instrumentation we did on the SR1500AL testbed is unique and the established experimental method is more convincing than simulation. We are able to do extensive experiments which bring several new insights.

The rest of this paper is organized as follows. Section 2 presents background and related work of this study. Section 3 discusses our design and implementation of the DRAM DTM schemes on real systems. Section 4 describes the experimental methodology and workloads. Section 5 analyzes the experimental results and finally Section 6 summarizes the study.

2. BACKGROUND AND RELATED WORK

Thermal Management in Computer Systems. Thermal management has become a major research focus in recent years. Most studies so far have focused on processors and hard disks in server systems and data centers. Brooks and Martonosi study different processor DTM mechanisms, which include scaling the clock frequency or voltage [1]. Skadron et al. develop a thermal model for individual functional blocks using thermal resistances and capacitances derived from the layout of the micro-architecture structures [19]. They further extend the model to HotSpot, which models thermal behavior at microarchitecture level using a network of thermal resistances and capacitances and can identify the hottest unit on chip [20]. They also propose several system-level DTM techniques; for example, migrating computation to underutilized hardware units from overheated ones. Li et al. study the thermal constraints in the design space of CMPs [14]. Donald and Martonosi explore the design space of thermal management techniques for multicore processors [3]. Regarding the DTM for the hard disk drives, Gurumurthi et al. develop models to capture the capacity, performance and thermal behavior of disk drives. They also present two DTM techniques for hard disks, exploiting the thermal slack or throttling disk activities [5]. Kim et al. further develop a performance-temperature simulator of disk drives and study the thermal behavior and management of storage systems using server workloads [13].

Another important and related area is applying DTM techniques at the system and data center level. Moore et al. use temperature-aware workload placement algorithm to reduce data center cooling costs [18]. Heath et al. propose Mercury, a temperature emulation suite for servers; they also develop and evaluate Freon, a system

for managing thermal emergency in server clusters [6]. Choi et al. propose ThermoStat, a CFD-based tool, to study thermal optimizations at run time in server systems as well as the layout optimization in design phase [2]. This study focuses on DRAM memory subsystems in individual servers. The proposed methods can be used along with the other methods.

Thermal Issue of DDR2 and FB-DIMM Memories. With the rapid increase of DRAM capacity and bandwidth, DRAM memory subsystem now consumes a significant portion of total system power. The DRAM memory has a general trend of power increase of 5-6% per year, as the bandwidth doubles every three years. The DRAM die size itself has been growing at a rate of 5-6% per year. In the latest generation server systems, DRAM power consumption can be comparable to that of processors. DRAM thermal problem has become a real issue recently for both DDR2 DRAM and Fully Buffered-DIMM (FB-DIMM). A recent study has reported that on a mobile system, the temperature of DDR2 DRAM devices may exceed their thermal design point of 85°C when running real workloads at an ambient temperature of 35°C [12]. On sever platforms, the recently deployed FB-DIMM has become a focus for DRAM thermal studies [15, 16].

FB-DIMM is designed to support both high bandwidth and large capacity by using narrow and high-frequency channels with point-to-point communication [4]. An AMB (advanced memory buffer) is placed into each DIMM for transferring data between the DRAM devices and the memory channel. In FB-DIMM, AMB is a hot spot because of its high power density (18.5 $Watt/cm^2$). Without thermal control, the temperature of AMB can exceed its thermal design point of 110°C. In practice, a product server may hold the AMB temperature under 100°C for safety. Additionally, the heat generated by AMB will spread to DDR2 DRAM devices with a lower thermal design point of 85°C, making them potential hot spots.

DRAM Thermal Behavior. As modeled in the Micron power calculator [17] and an Intel data sheet [9], the power consumption of a DRAM memory subsystem is almost linear to the memory throughput. It consists of two parts, static power which is mostly a constant (but configuration-dependent), and dynamic power which is almost proportional to memory throughput. If the memory throughput is kept at a constant level, the DRAM temperature will raise and stabilize in a few minutes. The stable temperature can roughly be defined as $T_{\text{stable}} = T_{\text{ambient}} + \sum P_{\text{i}} \times \Psi_{\text{i}}$, where T_{ambient} is the DRAM ambient temperature, Ψ_{i} is the thermal resistance between a component i in the system and the DIMM, and P_{i} is the power consumed by the component. Components in this equation include the DIMM itself and may include adjacent DIMMs and other subsystems like processor or hard disk if the thermal interaction between them and the DIMM is not negligible. In FB-DIMM, the power consumption of a typical AMB is in the range of 4-8 Watts; and that of a typical DDR2 DRAM device is about 5 Watts.

Dynamic Thermal Management Schemes for Memories. In practice, two DTM schemes have been used to prevent AMB or DRAM devices from overheating. In *thermal shutdown*, the memory controller (or the operating system) periodically reads the temperature of DIMMs from thermal sensors located on the DIMM and, if the reading exceeds a preset thermal threshold, stops all accesses to memory until the temperature drops below the specified threshold by a certain margin. In *bandwidth throttling* [8, 16], the memory controller gradually throttles memory throughput as temperature starts to rise to prevent it from crossing critical (shutdown) thermal threshold. The throttling is done by counting and limiting the number of DRAM row activations in a given time window.

Lin et al. [15] propose a dynamic DRAM thermal model as well as the DTM-ACG and DTM-CDVFS schemes and evaluate them using a simulator. The two schemes are discussed in detail in Section 3. We further extend this work in our paper and provide detailed evaluation and analysis of these schemes on real systems.

Other Related Work. Isci et al. [11] have proposed a runtime phase prediction method to manage mobile processor power consumption for memory intensive applications. They use DVFS during memory-bound phases of a workload to reduce processor power. By contrast, DTM-CDVFS is triggered in thermally constrained systems and the objective is to improve performance and reduce thermal heat exchange in addition to improving processor power efficiency; and the evaluation in this study has been done in a multi-core server system as opposed to a single-core mobile platform. Since memory temperature change is much slower than program phase change, thermal emergency is likely a more reliable trigger for DVFS with a performance target, though phase prediction can work when thermal emergency does not appear. Another study by Isci et al. [10] proposes methods to use per-core DVFS in managing the power budget of a multicore processor. Besides the difference that this study is focused on memory thermal management, per-core DVFS is not available on our platforms.

3. DESIGN AND IMPLEMENTATION ISSUES

In general, a DTM scheme can be divided two interdependent parts, *mechanism* and *policy*. The mechanism enforces DTM decisions made by the policy and also provides inputs to it while the policy decides when and what thermal actions to trigger. The goal of a DTM policy is to prevent memory from overheating or going above its maximum safe operating temperature. This is accomplished by continuously monitoring memory temperature and forcing memory to reduce its power by putting a cap on memory throughput when the temperature crosses a predefined threshold called T_{critical} (Figure 1). The bandwidth allowed at this point is drastically limited to protect that part, leading to significant degradation in system responsiveness and performance.

To smooth the effects of thermal management and reduce its impact on system performance, a well designed DTM policy may try to reduce memory bandwidth more gracefully when memory temperature starts to approach a critical threshold [19, 15]. To accomplish this, a DTM policy usually defines another threshold called T_{tm} where an adaptive DTM mechanism, which supports multiple running levels, starts to get activated. The range $(T_{\text{tm}}, T_{\text{critical}})$ is normally broken into thermal zones with each thermal zone having an associated set of actions that are designed to lower memory temperature. Thermal zones with higher temperatures trigger more aggressive actions necessary to bring memory temperature within the T_{tm} threshold. Note that if temperature ever crosses T_{critical} threshold and reaches T_{shutdown} point, the memory controller will shut down the memory subsystem to avoid any physical damage to that part, though this should never happen in a properly designed system with bandwidth throttling.

3.1 Memory DTM Mechanisms

A memory DTM mechanism should generally consist of three components: a memory temperature monitor or estimator, a DTM policy trigger, and a method for controlling memory temperature. We have designed different mechanisms to support four thermal management policies, namely DTM-BW, DTM-ACG, DTM-CDVFS, and DTM-COMB, and implemented them on two Linux servers with Intel Xeon 5160 processors. Each DTM mechanism is an integration of hardware/software components that together provide the required functions and capabilities to support DTM policy.

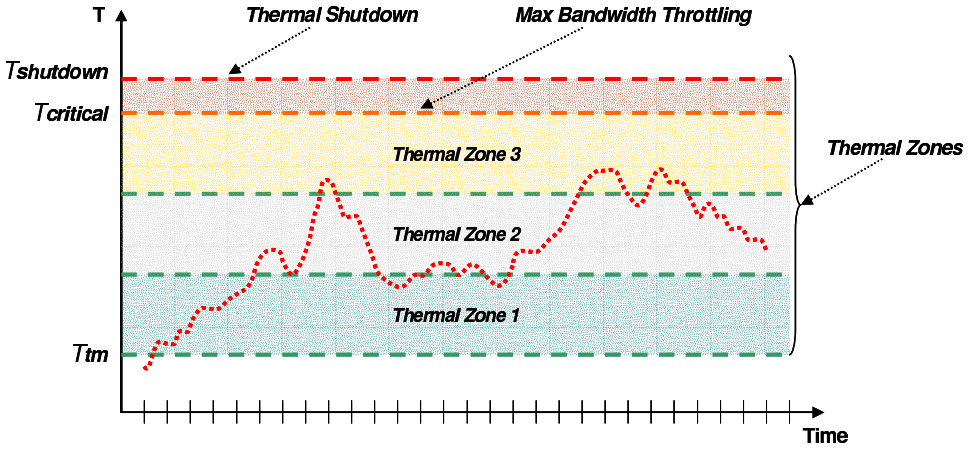


Figure 1: The idea of using thermal zone to guide the design of memory DTM policies. Thermal emergency level, which is used in the discussions of our specific DTM implementations, is the same concept.

Temperature Monitoring. A DTM scheme makes thermal management decisions based on current (and possibly past or predicted future) memory temperature. The temperature can be either measured if thermal sensors are available or conservatively predicted if otherwise. Temperature readings at multiple points should be obtained because a memory subsystem may have multiple hot spots, where the temperature may potentially cross the critical thermal point of that part if no thermal solution is used. Both of our servers use FB-DIMM based memory subsystems and by their configurations the AMBs are hot spots[1]. Thermal sensors are located in the AMB of each FB-DIMM. Thus, the AMB temperature can be directly measured and used for making DTM decisions. In addition, our SR1500AL system hosts multiple temperature sensors that measure the front panel temperature (system ambient), CPU inlet temperature, memory inlet temperature and memory exhaust temperature. Those temperature readings do not affect DTM decisions but allow us to get a detailed thermal profile during program execution.

Policy Trigger. A DTM policy needs to periodically check whether any of the thermal thresholds have been crossed and invoke thermal control logic if necessary. We implement the DTM policy as a monitoring utility, which is periodically woken up by the OS scheduler. We used a default interval of one second in our experiments. Since DRAM thermal constants are large (it takes roughly a few hundred seconds for DRAM devices or AMBs to reach their TDPs from idle state), one-second interval is adequate to trigger thermal management actions and protect memory from overheating. It is also sufficiently long to avoid any noticeable performance overhead from the monitor itself.

Memory Thermal Control Methods. When a thermal threshold is crossed and the event is detected by the monitoring utility, some actions need to be taken to lower memory temperature. Since DRAM power and therefore temperature are closely related to memory throughput (with all other conditions such as airflow and inlet temperature being equal), temperature can be lowered by reducing memory throughput. We have used three approaches that control memory activity either from the memory side or the processor side. The first approach called *Bandwidth Throttling* sets a limit on the number of memory accesses that are allowed in a certain time window. Intel 5000X chipset used in both of out servers allows clamping the number of DRAM row activations over a specified time window [8]. The DTM-BW policy uses this capability to throttle memory throughput at different levels based on current thermal zone. The second approach called *Core Gating* reduces memory throughput by limiting the number of active cores through CPU hot plug module in the Linux kernel (version 2.6.20). When a CPU is unplugged it is logically removed from the OS and placed into a sleep state by executing a halt instruction. Note that the overhead of this operation is very small given one-second granularity of our DTM policies. The last approach uses the feature of processor *Voltage and Frequency Scaling* to reduce memory throughput. The Xeon 5160 processors used in our servers support four different (Frequency, Voltage) operating points: (3.0 GHz, 1.2125 V), (2.667 GHz, 1.1625 V), (2.333 GHz, 1.1000 V), and (2.000 GHz, 1.0375 V). In the latter two approaches, bandwidth throttling is enabled when memory temperature is close to its thermal threshold to avoid any possibility of overheating.

[1]DRAM devices can be hot spots in other configurations of FB-DIMM.

3.2 Memory DTM Polices

The objective of a DTM policy is to maximize the overall system performance without crossing the thermal threshold of any part of the system. A secondary objective is to improve the overall system power efficiency. In general, DTM policies have to reduce memory throughput to lower memory temperature, but their strategies can be different as discussed above. It is important to note these DTM policies do not guarantee a certain level of memory temperature or activity. Rather, they change memory or processor operating parameters with the aim of reducing memory bandwidth while increasing system's ability to better tolerate constrained thermal environments. If these policies are not effective in reducing memory temperature, they will all enable bandwidth throttling as a safeguard when memory temperature crosses a critical T_{critical} threshold as shown in Figure 1.

Thermal Emergency Level and Running Level. As discussed earlier, a general approach in our DTM policy design is to quantize memory temperature into *thermal zones* or *thermal emergency levels* [19] and associate each level with a system *thermal running level*. The use of running levels has appeared in real systems, i.e. in bandwidth throttling of Intel chipset 5000X [8]. In this study, system running levels are defined in terms of processor frequency, number of active cores and allowed memory bandwidth.

In general, a thermal running level with better system performance also generates more heat. Every time a policy module is executed, it reads the temperature sensors, determines the thermal emergency level, and then decides the thermal running level for the next time interval. If the new running level is different from the current one, a thermal action will be taken to change the thermal running level.

Table 1 describes the settings of the thermal emergency levels and the thermal running levels for the two servers. It is important to note that the number of the emergency levels and that of the running levels do not have to equal. The number of thermal zones is based on the inherent relationship between DRAM temperature and memory bandwidth while the number of thermal running levels is based on the underlying HW capabilities. For example, there could be more than four running levels if the two processors are quad-core. Also, it is a coincidence that DTM-ACG and DTM-CDVFS have the same number of running levels.

We used the following methodology to define thermal emergency level for our two systems. On the Intel SR1500AL we set T_{critical} threshold to 98°C. This threshold is based on a conservative 100°C AMB thermal threshold with a 2-degree margin to ensure safe operation of the system. The Intel SR1500AL is put into a hot box and the system ambient temperature is set to 36°C, which emulates a constrained thermal environment. Four thermal emergency levels are then defined in decrements of 4 degrees: [94, 98), [90, 94), [86, 90), [-, 86). The PE1950 is located as a standalone box in an air-conditioned room with a system ambient temperature of 26°C. To better understand the thermal behavior of such a server with an ambient temperature of 36°C, we artificially lower the 100°C AMB thermal threshold by 10°C and set T_{critical} to 88°C accordingly. The thermal emergency levers for PE1950 are then defined in decrements of 4 degrees similar to the SR1500AL system: [84, 88), [80, 84), [76, 80), [-, 76). Note that in both systems the lowest thermal emergency level does not impose any constraints and allows for full system performance. For safety concern, the chipset's bandwidth throttling is enabled when the system is in the highest thermal emergency level to ensure that overheating will not happen.

DTM-BW Policy. This policy only performs bandwidth throttling. It resembles the bandwidth throttling in Intel chipset 5000X [8]; and we use it as a reference to evaluate other DTM policies. It uses the bandwidth limiting function of the chipsets, which are available in both systems, to cap the bandwidth usage according to the current thermal emergency level. Setting the limit to 2GB/s on the PE1950 will guarantee that the memory will not overheat; and so does using the 3GB/s limit on the SR1500AL. As Table 1 describes, four thermal running levels are used. The limits are enforced in the chipset by limiting the number of DRAM row activations in a time window. Because close page mode is used, bandwidth usage is mostly proportional to number of DRAM row activations. The default window of 66ms is used, which is suggested by the chipset designers.

DTM-ACG Policy. This policy uses core gating to indirectly reduce memory traffic. It is based on the observation that when a subset of cores is disabled, the contention for last level cache from different threads is reduced, leading to reduced memory traffic. If the workload is bandwidth-bound, then system performance will be improved. Note that for a memory-intensive workload, the memory thermal constraint puts a limit on memory bandwidth that the program execution may utilize. Therefore, the gain from memory traffic reduction can more than offset the loss of computation power from core gating. Both servers have two dual-core processors. We always retain one active core per processor to fully utilize their caches to reduce memory traffic. Therefore, thermal running levels associated with the last two thermal emergency levels support the same number of active cores. The main difference between them is that in L4, as with other DTM policies described in this section, maximum memory bandwidth throttling is also enforced. Note that when one or more cores are gated the threads allocated to these cores get migrated to active cores resulting in serialized sharing of core and cache resources with other threads. We found that the default scheduling interval of 100ms of the Linux kernel works well, containing cache thrashing while maintaining fairness and smoothness in process scheduling.

DTM-CDVFS Policy. This policy uses processor DVFS to indirectly throttle memory traffic. The main source of performance improvement, as to be showed in Section 5, is the reduced processor heat generation. Lowering processor voltage and frequency leads to reduced processor power consumption and heat exchange with other components, including memory. Consequently, the memory subsystem may run at high speed for longer time than normally allowed. This effect has been observed on both servers, but is more obvious on the Intel SR1500AL because the processors are located right in front of FB-DIMMs on the airflow path in that system. The four running levels under this policy correspond to four frequency and voltage settings of Xeon 5160 CPU. In this paper, DTM-CDVFS policy performs frequency/voltage transitions on all processors and cores simultaneously and uniformly. A policy that supports such differentiation is worth further investigation but we leave it to future work.

DTM-COMB Policy. DTM-COMB combines DTM-ACG and DTM-CDVFS by integrating core gating and coordinated DVFS. The goal of this policy is to take full advantage of the combined benefits offered by DTM-CDVFS' larger thermal headroom and reduced memory traffic enabled by core gating in DTM-ACG. Table 1 shows four thermal running levels defined for this policy by combining the number of active cores and processor operating frequency. Similar to DTM-ACG, we keep at least one core active for each CPU in DTM-COMB to fully utilize the caches.

4. EXPERIMENTAL METHODOLOGY

4.1 Hardware and Software Platforms

We conducted our experiments on two small-scale servers as standalone systems. Both machines use FB-DIMM and the memory hot spots are AMBs, therefore we are only concerned with AMB temperatures thereafter. (DRAM devices can be hot spots in different configurations.) The first one, PE1950, is a Dell PowerEdge 1950 1U server put in an air-conditioned room as a standalone system. It has Intel 5000X chipset and two dual-core, 3.0GHz Intel Xeon 5160 processors. Each has a shared, 4MB, 16-way set associative L2 cache; and each core of the processor has a private 32KB instruction cache and a private 32KB data cache. The machine has two 2GB 667MT Fully Buffered DIMM (FB-DIMM) as the main memory. The second machine is an Intel SR1500AL machine which is instrumented for thermal and power study. It has almost the same configuration as the PE1950 except that it has four 2GB 667MT FB-DIMM. On the SR1500AL, we are able to measure the power consumption of FB-DIMM and processors and processor exhaust temperature, which is also the memory ambient temperature on this system. The instrumentation on the SR1500AL altered the air flow inside the machine, making it less stronger than as in a production machine. The reported temperature readings should not be assumed as on a product machine.

As mentioned before, we are more interested in the thermal behaviors of those machines in a constrained thermal environment.

Thermal Emergency Level	Machine	L1	L2	L3	L4
AMB Temp. Range (°C)	PE1950	(-, 76.0)	[76.0, 80.0)	[80.0, 84.0)	[84.0 88.0)
AMB Temp. Range (°C)	SR1500AL	(-, 86.0)	[86.0, 90.0)	[90.0, 94.0)	[94.0 98.0)
Thermal Running Level	**Machine**	**L1**	**L2**	**L3**	**L4**
DTM-BW: Bandwidth	PE1950	No limit	4.0GB/s	3.0GB/s	2.0GB/s
DTM-BW: Bandwidth	SR1500AL	No limit	5.0GB/s	4.0GB/s	3.0GB/s
DTM-ACG: # of Active Cores	Both	4	3	2	2
DTM-CDVFS: Frequency	Both	3.00GHz	2.67GHz	2.33GHz	2.00GHz
DTM-COMB:: # of Cores@Frequency	Both	4@3.00GHz	3@2.67GHz	2@2.33GHz	2@2.00GHz

Table 1: Thermal emergency levels and thermal running levels.

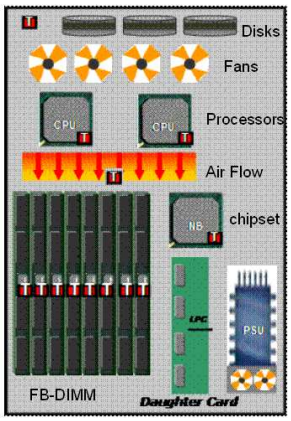


Figure 2: Intel SR1500AL system with thermal sensors ("T").

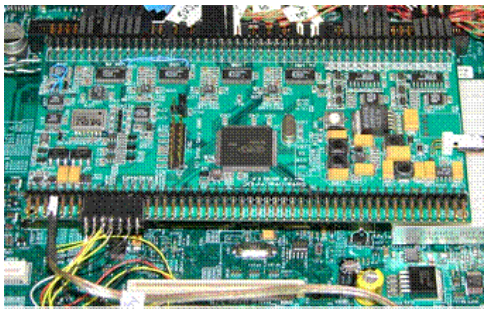


Figure 3: The daughter card.

To emulate such an environment, we put the Intel SR1500AL into a hot box, which allows us to set the system ambient temperature to 36°C, assuming a room temperature of 26°C and an arbitrary 10°C increase in system ambient temperature. We put the PE1950 into an air-conditioned room of temperature 26°C and artificially lower the thermal design point of AMB by 10°C in the implementation of DTM policies. We use the two different machines to crosscheck our experiment results, and the SR1500AL allows us to evaluate power and energy savings.

Figure 2 shows a system diagram of the SR1500AL server. We instrumented the Intel SR1500AL with sensors that measure voltage, current and temperature of different system components. The analog signals from the power and thermal sensors are routed to a custom designed daughter card that hosts an array of A/D converters and low pass filters. The daughter card is shown in Figure 3. The data from A/D converters is sampled by a micro-controller that stores all the digital sensor data in a local buffer. The daughter card is connected to the host system through a LPC (low pin count) bus. We have implemented a user space application that accesses the daughter card using Linux LPC driver. The application reads sensor data periodically and stores it to a log file. In all experiments in this paper we used a sampling rate of once per 10ms. This sampling rate is sufficient given AMB thermal constants and time scales of thermal management mechanisms evaluated in our studies. The sample data are buffered inside the daughter card until the buffer is full; then they are transferred to the system memory. We have done an extensive evaluation to calibrate the sensors and ensure that sampling application does not introduce any overhead or artifacts in our measurements. We have run benchmarks and synthetic workloads with and without our sampling application and have never observed any measurable impact on their performance or system power consumption.

The two machines use the Red Hat Enterprise Linux 4.0 with kernel 2.6.20.3. Performance data are collected by `pfmon` using perfmon kernel interface and `libpfm` library [7]. We enable the CPU hot plug/remove functionality of the kernel to support core gating. Three types of performance statistics are collected using hardware counters: numbers of retired uops, L2 cache accesses, and L2 cache misses. We use the per-thread mode of `pfmon` to collect statistics for each benchmark. As discussed in Section 3, for DTM-ACG, when one core on dual-core processor is shut down, two programs will share the remaining core in a round-robin fashion. The time slice for the sharing is 100ms by default Linux kernel. We also perform a sensitivity analysis by varying the time slice and the result will be shown in Section 5.

4.2 Workloads

We run multiprogramming workloads constructed from the SPEC CPU2000 benchmark suite[2]. The applications are compiled with Intel C++ Compiler 9.1 and Intel FORTRAN Compiler 9.1. When the four-core machines run four copies of a same application, thirteen applications of SPEC CPU2000 reach higher AMB temperature than others: *wupwise, swim, mgrid, applu, vpr, galgel, art, mcf, equake, lucas, fma3d, gap* and *apsi*. Twelve out of the thirteen applications coincide with those selected by the simulation-based study [15]. The only exception is *gap*. To simplify the comparison between this work and the previous study, we do not include *gap* in our experiments. Then we constructed eight multiprogramming workloads from these selected applications as shown in Table 2. We ran all workloads twice and the differences in execution time are negligible. The results of a single set of experiments are reported.

Some SPEC applications had more than one reference inputs.

[2]We have also run partial experiments on workloads constructed from SPEC CPU2006.

Workload	Benchmarks
W1	swim, mgrid, applu, galgel
W2	art, equake, lucas, fma3d
W3	swim, applu, art, lucas
W4	mgrid, galgel, equake, fma3d
W5	swim, art, wupwise, vpr
W6	mgrid, equake, mcf, apsi
W7	applu, lucas, wupwise, mcf
W8	galgel, fma3d, vpr, apsi

Table 2: Workload mixes.

For those applications, we run all inputs and count them as a single run. In order to observe the long term memory temperature characteristics, we run the multiprogramming workloads as batch jobs. For each workload, its corresponding batch job mixes ten runs of every application contained in the workload. When one program finishes its execution and releases its occupied processor, a waiting program is assigned to the processor in a round-robin way. It is worth noting that, at the end of the batch job, there is small fraction of period that less than four applications running simultaneously. We observed that the fraction was less than 5% of total execution time on average.

We do not study DTM-TS (Thermal Shutdown) in this work for the following reasons. First, DTM-TS is a special case of DTM-BW. Second, it abruptly shuts down the whole system and makes system not running smoothly.

5. RESULTS AND ANALYSIS

In this section, we first briefly describe the DRAM thermal emergency observed on the servers. We then present the performance results of the four DTM policies, analyze the sources of performance gain, discuss the results of power saving and finally study the sensitivity of parameter selections.

5.1 Experimental Observation of DRAM Thermal Emergency

We present our observation of AMB temperature changes on the two server systems. Figure 4 shows the AMB temperature changing curves on the the SR1500AL when it runs homogeneous workloads described as follows. The machine has open-loop bandwidth throttling enabled by default in the chipset. We disable this function for AMB temperature below 100°. During the close-to-overheating periods (>100°C), the function is enabled to limit the memory bandwidth under 3GB/s for safety concern. We run four copies of each program on the four cores (on two processors) simultaneously. For each program, we run a job batch with twenty copies in total to observe the AMB temperature changes and report the results of the first five hundred seconds. The data are collected every one second. The server has four DIMMs and the highest AMB temperature among the four DIMMs is shown (most time the third DIMM has the highest AMB temperature).

Temperature changes of five selected programs are reported in the figure. Among them, *swim* and *mgrid* are memory intensive, and the other three are moderately memory intensive. Initially the machine is idle for a sufficiently long time for the AMB temperature to stabilize (at about 81°C). As it shows, with *swim* and *mgrid* the temperature will reach 100° in about 150 seconds. Then it fluctuates around 100°C because of the bandwidth throttling for machine safety. We have similar observations for other memory intensive programs (not shown). The other three programs, namely

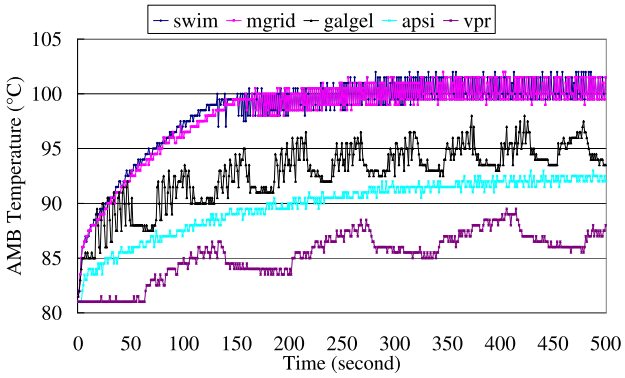


Figure 4: AMB temperature curve for first 500 seconds of execution.

galgel, apsi and *vpr*, are less memory intensive. Their temperatures rises in similar curves and then the temperature change patterns stabilize under 100°C.

Figure 5 shows the average AMB temperatures of the PE1950 when it runs the same homogeneous workloads. Unlike Figure 4, Figure 5 does not show memory overheating; instead, it shows how overheating would have happened for those workloads if the ambient temperature is high enough and no DTM is used. The PE1950 is put in a room with good air conditioning. (It also has a different cooling package.) Therefore, we are able to run memory-intensive workloads without having the system to overheating the AMBs (or the DRAM devices). Additionally, the server currently includes only two DIMMs. If four DIMMs were used, as we observed on the SR1500AL, the AMB temperature would be significantly higher than reported. Only the AMB temperature of the first DIMM is shown because it always has higher temperature than the other one. The temperature sensors have noises which appear as high spikes in temperature readings (which is visible in Figure 4), therefore we exclude 0.5% sampling points with the highest temperatures to remove those spikes.

We have following observations. First, average AMB temperature varies significantly across those homogeneous workloads. Ten programs have average AMB temperature higher than 80°C: *wupwise, swim, mgrid, applu, art, mcf, equake, facerec, lucas* and *fma3d*. As shown in the previous study [15] and confirmed in our experiments using performance counters, these ten programs have high L2 miss rates. Consequently, they have higher memory bandwidth utilization, higher memory power consumption and therefore higher AMB temperatures than the other workloads. Four programs, namely *galgel, gap, bzip2* and *apsi*, have moderate memory bandwidth utilization and their average AMB temperatures range between 70°C and 80°C. The other twelve programs have small memory bandwidth utilization and their AMB temperatures are below 70°C. Second, there are big gaps between the average and the highest AMB temperatures. We have found the main reason is that it takes a relatively long initial time, around two hundred seconds, for AMB to reach a stable temperature. Additionally, for some workloads, the AMB temperatures keep changing due to the program phase changes in their lifespan.

5.2 Performance Comparison of DTM Polices

Figure 6 compares the performance of the four DTM policies and a baseline execution with no memory thermal limit on the two servers. As discussed in Section 4, on the PE1950, we use an artificial thermal threshold of 90°C to reveal the impact of memory thermal limit. The no-limit experiments are done without enforc-

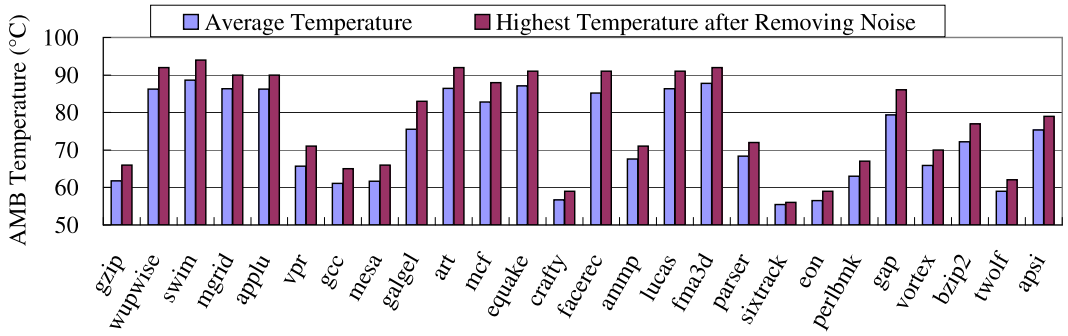


Figure 5: AMB temperature when memory is driven by homogeneous workloads on the PE1950 without DTM control.

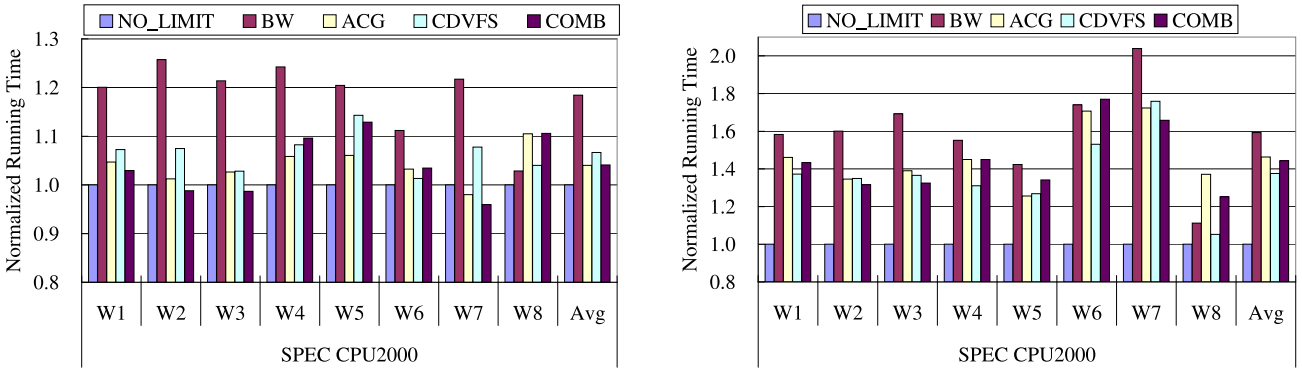


Figure 6: Normalized running time.

ing that artificial TDP. On the SR1500AL, we are able to control the ambient temperature, so we run the no-limit experiments with an ambient temperature of 26°C and run the other experiments with an ambient temperature of 36°C. We disable the built-in bandwidth throttling feature of the chipset in the no-limit experiments.

We have the following observations for workloads from SPEC CPU2000. First of all, the results confirm that the use of simple bandwidth throttling (DTM-BW) may severely downgrade the system performance. On average, the performance degradation is 18.5% on the PE1950 and 59.3% on the SR1500AL. Our detailed statistics show that there is a strong correlation between the memory bandwidth utilization and the performance degradation. For example, all workloads except $W5$ and $W8$ have larger than 50% slowdown with DTM-BW, while the slowdowns for $W5$ and $W8$ are 42.3% and 11.3%, respectively, on the SR1500AL. The performance counter data show that $W8$ has 17.3 L2 cache misses per microsecond, which is the lowest among the eight workloads. This means it is less memory-intensive than others.

Second, the results also confirm that DTM-ACG may significantly improve performance over DTM-BW. On average, DTM-ACG improves the performance of CPU2000 workloads by 13.3% on the PE1950 and 7.2% on the SR1500AL. The maximum improvement is 24.2% and 21.7%, respectively. In comparison, the previous simulation-based study [15] reports an average improvement of 16.3% using the same workloads. The main source of improvement comes from the reduction on L2 cache misses, which will be detailed in Section 5.3. As for the difference in the results from the two servers, several factors may contribute to it, including the differences in cooling package, memory bandwidth, ambient temperature, and the layout of the processors and DIMMs on motherboard. We also observe performance degradation of DTM-ACG over DTM-BW on workload $W8$, which is 7.4% on the PE1950 and 23.2% on the SR1500AL, respectively. This scenario was not reported in the previous study. As to be shown in Figure 7, DTM-ACG actually reduces the L2 cache misses of $W8$ by 6.4% on the PE1950 and 7.4% on the SR1500AL. We believe that for this workload the DTM-ACG policy may stop processor cores too proactively. This is not a fundamental problem of the policy, but indicates that the policy may be further refined for certain types of workloads.

Regarding DTM-CDVFS, we have surprising findings that are very different from the previous study. On average, DTM-CDVFS may improve performance over DTM-BW by 10.8% on the PE1950 and 15.3% on the SR1500AL. By contrast, the previous study reports only 3.4% average improvement. It is also remarkable that the scheme improves the performance of every program on SR1500AL, ranging from 5.7% to 23.9%. On PE1950, the maximum improvement is 18.0% and only $W8$ has small performance degradation of 1.1%. The main reason behind the performance improvements, as to be discussed in details in Section 5.3, is related to the thermal interaction between the processors and the memory. The previous study did not consider the heat dissipation from the processor to the memory. As the results indicate, that factor should be significant in the DRAM thermal modeling and cannot be ignored. In fact, the performance improvement is larger on the SR1500AL than on the PE1950 because on its motherboard the processors are physically closer to the DIMMs. We will present more experimental results from the SR1500AL to support this finding.

We have also run two workloads from SPEC CPU2006 on PE1950, $W11$ with applications *milc*, *leslie3d*, *soplex* and *GemsFDTD*, and

$W12$ with *libquantum*, *lbm*, *omnetpp* and *wrf*. The findings for workloads from CPU2000 still hold for them. DTM-BW degrades the performance by 21.4% and 25.4% for $W11$ and $W12$ when compared with no-limit, respectively. DTM-ACG improves performance by 7.6% and 14.4% when compared with DTM-BW, respectively. DTM-CDVFS has better performance for both workloads, improving performance by 16.8% and 17.8% over DTM-BW on the two servers, respectively.

The performance of DTM-COMB is very close to that of DTM-ACG on average on both machines. On average for SPEC CPU2000 workloads, the performance of DTM-COMB is degraded by 0.1% on PE1950 and improved by 1.4% on SR1500AL, compared with DTM-ACG. The DTM-COMB may improve performance up to 5.7% (for $W12$ from SPEC CPU2006). It is remarkable that DTM-COMB can improve performance for $W2$, $W3$ and $W7$ on PE1950, when compared with no-limit. This is possible because we observe that for some programs, the L2 cache miss rate decreases sharply when running alone as shown later in Section 5.3.

5.3 Analysis of Performance Improvements by Different DTM Policies

In this section, we analyze the sources of performance improvements by DTM-ACG, DTM-CDVFS and DTM-COMB when compared with DTM-BW.

Reduction of L2 Cache Misses. It has been reported in the previous study [15] that the improvement by DTM-ACG is mostly from the reduction of memory traffic, which is from the reduction of L2 cache misses: When the shared L2 cache is used by fewer programs, cache contention is reduced and thus there will be fewer cache misses. The previous study collected memory traffic data to demonstrate the correlation. On our platforms, we can only collect the number of L2 cache misses. The total memory traffic consists of cache refills from on-demand cache misses, cache write-backs, memory prefetches, speculative memory accesses, and other sources including cache coherence traffic. Nevertheless, cache refills are the majority part of memory traffic, therefore the number of L2 cache misses is a good indication of memory traffic.

Figure 7 shows the normalized number of L2 cache misses on both machines. We have several observations from the data. First, the number of L2 cache misses changes very slightly by using DTM-BW when compared with no-limit. This is expected because the number of on-demand L2 cache misses should have virtually no change when memory bandwidth is throttled. Second, the total number of L2 cache misses does decrease significantly by DTM-ACG, compared with that of DTM-BW. The reduction is up to 35.2% and 40.7% on the PE1950 and the SR1500AL, respectively. The average reduction are 26.8% and 29.3%, respectively. The result confirms the finding of the previous study that DTM-ACG reduces L2 cache misses significantly. On the other hand, DTM-CDVFS does not cause any visible changes of the total number of L2 cache misses, while the previous study reported memory traffic may be reduced due to the reduction of speculative memory accesses. The difference is likely related to differences in the processor models, particularly how many outstanding memory accesses are allowed and whether a speculative memory instruction is allowed to trigger an access to the main memory. The DTM-COMB has very similar L2 cache miss reduction as DTM-ACG. The average reductions are 24.8% and 30.1% on the PE1950 and the SR1500AL, respectively.

Reduction of Memory Ambient Temperature by DTM-CDVFS. As discussed earlier, the performance of DTM-CDVFS is comparable to that of DTM-ACG. In fact, it is visibly better than DTM-ACG on the SR1500AL. This is a surprise finding: the previous study reports that DTM-CDVFS has only slight performance advantage over DTM-BW; and the main benefit of DTM-CDVFS was improved system power efficiency. We speculated that the processor heat generation has an impact on the memory DIMMs, which was ignored in the thermal modeling of the previous study. If the processor is physically close enough to the DIMMs, then the heat dissipation from the processor may further increase the DIMM ambient temperature. Consequently, the DIMMs may overheat more frequently than predicted by the thermal model in the previous study. Since DTM-CDVFS improves the processor power efficiency, it may reduce the heat generation from the processor and therefore alleviate the problem, which will improve memory bandwidth utilization. If that is a significant factor, then the observed performance improvement can be explained.

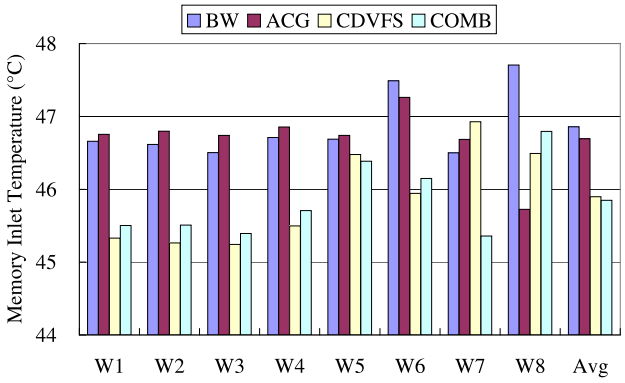
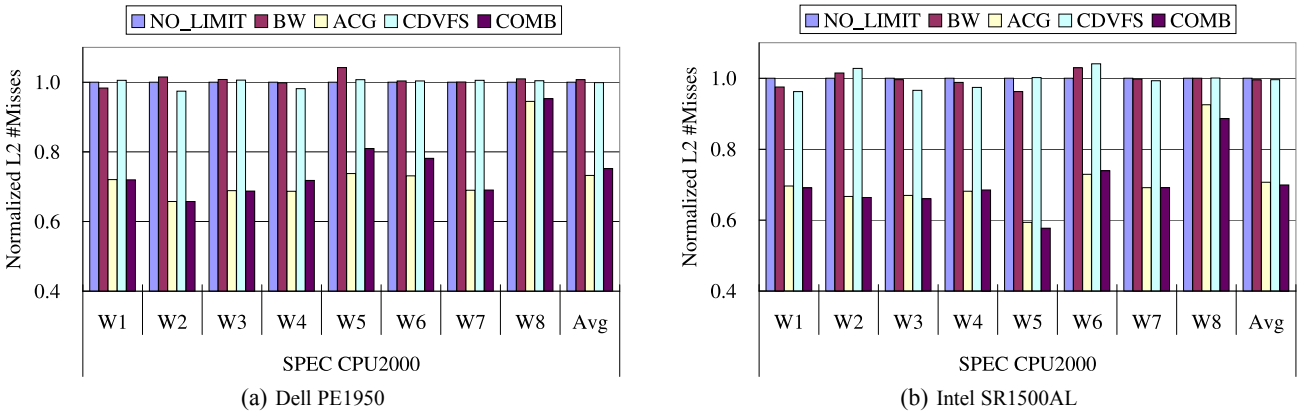


Figure 8: Measured memory inlet temperature.

To confirm the above theory, we have looked into the inside of each machine. On both machines the processors and the DIMMs share the same set of cooling fans, and the air flow first passes the processors then the DIMMs. The processor and DIMMs are slightly misaligned along the cooling air flow on the PE1950. On the SR1500AL, one of the two processors is aligned with the DIMMs along the cooling air flow. Additionally, the distance between the processors and the DIMMs is as close as about 5cm.

We collect the temperature readings through a sensor put on the air path between the processors and the DIMMs inside the SR1500AL. Such a sensor is not available on PE1950. The data show a strong correlation between the memory inlet temperature difference and the performance improvement of DTM-CDVFS over DTM-BW. Figure 8 compares the average temperature of the four DTM schemes. The system ambient temperature of the SR1500AL is set to 36°C. As the figure shows, the cooling air is heated up by about 10°C when it reaches the memory. The processor exhaust (memory inlet) temperature is visibly lower with DTM-CDVFS or DTM-COMB than with DTM-BW or DTM-ACG for workloads $W1$ to $W6$. As to be discussed in Section 5.4, DTM-CDVFS and DTM-COMB reduce processor power consumption but DTM-ACG does not when compared with DTM-BW. Workloads $W7$ and $W8$ are exceptions: for $W7$ the temperature is slightly higher with DTM-CDVFS than with the other schemes; and for $W8$ it is between DTM-BW and DTM-ACG. On average, the temperature is 46.9°C, 46.7°C, 45.9°C and 45.8% with DTM-BW, DTM-ACG, DTM-CDVFS and DTM-COMB, respectively. The maximum difference is 1.6°C. We highlight that we carefully calibrated those sensors and that random sensor noises do not affect the accuracy of average temperature. Those differences seem to be small; however, the range of working temperatures of memory intensive workloads is less than 20°C on that



(a) Dell PE1950 (b) Intel SR1500AL

Figure 7: Normalized numbers of L2 cache misses.

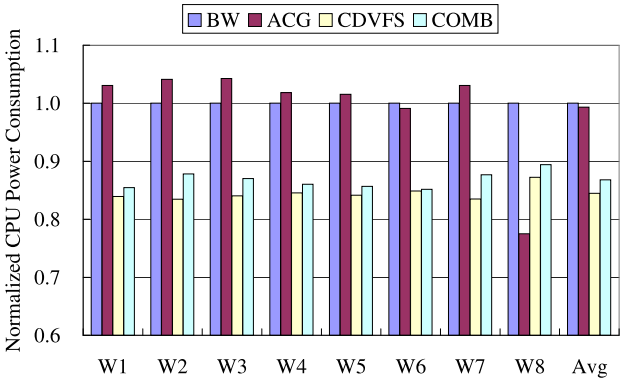


Figure 9: CPU power consumption.

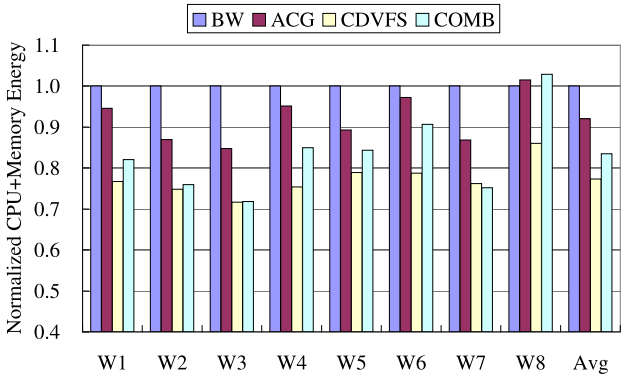


Figure 10: Normalized energy consumption of DTM policies.

server as shown in Figure 4. Therefore, a one-degree difference can have a noticeable impact on performance. The result strongly suggests that the layout design of server inside should give more attention to the memory subsystem.

5.4 Comparison of Power and Energy Consumption

On the SR1500AL we are able to measure the power consumption of individual system components including the processors, DIMMs, system fans and other components.

Power Consumption of Processors and DIMMs. We are only interested in the power consumption of the processors and DIMMs because for our workloads the power consumption of the other components is almost constant. The processors consume slightly more than a third of the system power; and the DIMMs consume slightly less than a third. In our experiments, we also found that the power consumption of the DIMMs is very close for all workloads except workload $W8$, which is less memory intensive than the others. Part of the reason is that static power is a large component of FB-DIMM power. Therefore, we only compare the processor power consumption.

Figure 9 shows the average power consumption with different DTM policies. The data are normalized to those of DTM-BW. As expected, DTM-CDVFS and DTM-COMB consume less processor power than the other two policies. On average, the processor power consumption of DTM-CDVFS and DTM-COMB is 15.5% and 13.2% lower than that of DTM-BW, respectively. There is a very small difference between the power consumption by DTM-BW and DTM-ACG. This is mainly due to the fact that the latest generation processors are packed with a number of energy efficient features. They apply extensive clock gating to idle functional blocks when processors are stalled by the long-latency memory accesses. Thus, for memory-intensive workloads with frequent last level cache misses, most functional components in the processor core have already been clock-gated yielding little additional benefit from gating the entire core.

Energy Consumption. Figure 10 shows the total energy consumption of processors and memory. All values are normalized to those of DTM-BW. On average, compared with DTM-BW, DTM-ACG, DTM-CDVFS and DTM-COMB can save energy by 6.0%, 22.0% and 16.5%, respectively. The energy saving of DTM-ACG comes from the reduction of running time because its power consumption is very close to that of DTM-BW. The energy savings for DTM-CDVFS and DTM-COMB come from both power saving and reduction of running time.

5.5 Sensitivity Analysis of DTM Parameters

Ambient Temperature. The performance of DTM policies shown in Section 5.1 on the SR1500AL is from the experiments with a system ambient temperature of 36°C and an AMB TDP of 100°C. We also have run experiments on SR1500AL with a lower system ambient temperature of 26°C and with an artificial AMB thermal threshold of 90°C. This setting is the same as that used on the PE1950, and has the same gap (64°C) between the ambient

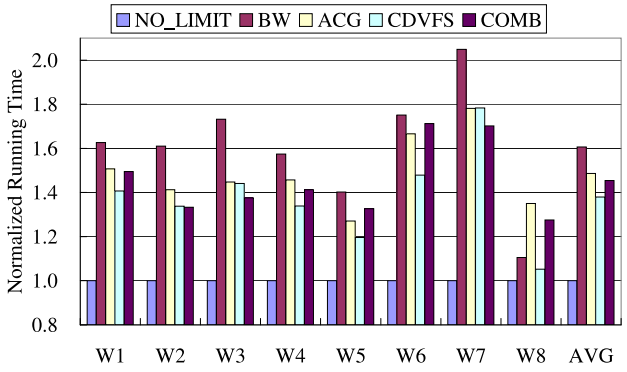


Figure 11: Normalized running time on Intel SR1500AL at a room system ambient temperature (26°C).

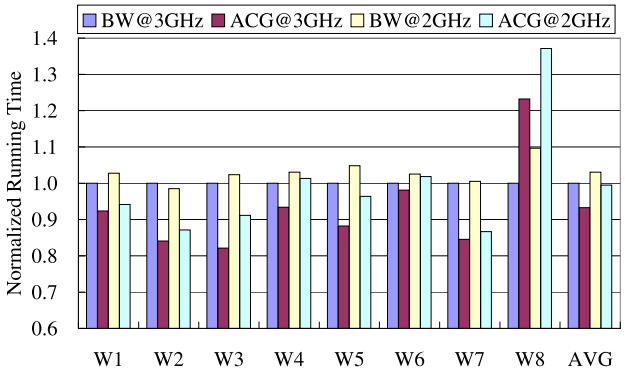


Figure 12: Comparison of performance between DTM-ACG and DTM-BW under two different processor frequencies on Intel SR1500AL.

temperature and the TDP temperature as the first set of experiments on the SR1500AL. The experiment has two purposes. First, by keeping the temperature gap the same while changing the ambient temperature, the new result will help understand how the ambient temperature affects performance. Second, because the performance improvements are different on the two servers, the new result may reveal whether the difference is related to their differences in ambient temperatures.

Figure 11 compares the performance of four policies on SR1500AL in the new setting. It indicates that the performance is very similar to that on the same machine with higher system ambient temperature of 36°C. On average, DTM-BW degrades performance by 60.6% over no-limit. The degradation is 59.3% with the higher ambient temperature. On average, DTM-ACG and DTM-CDVFS improve performance by 8.1% and 16.4% over DTM-BW, respectively. The improvements are 7.2% and 15.3% with an ambient temperature of 36°C (Figure 6(b)), respectively. The performance comparison regarding individual workload are also similar. The similarity indicates that the performance of DTM schemes is strongly correlated to the gap between the ambient temperature and AMB TDP.

Processor Frequency. In previous experiments, we run processor cores at full speed (3.0 GHz) for DTM-BW and DTM-ACG. We also want to see what happens if a lower processor speed (2.0 GHz) is used. Figure 12 compares the performance with two processor speeds for DTM-BW and DTM-ACG on the SR1500AL. First, on average, the performance with the lower processor speed is degraded by 3.0% and 6.7% compared with that with the higher speed for DTM-BW and DTM-ACG, respectively. We find that the less memory-intensive workload $W8$ has larger performance degradation than the others. This is expected since the performance of compute-intensive workloads is more sensitive to processor frequency. Isci et al. also present that the performance degradation is small for memory-intensive workloads with low frequency mode [11]. If $W8$ is excluded, the performance degradation is only 2.1% and 5.8% for DTM-BW and DTM-ACG, respectively. Second, DTM-ACG improves performance similarly under both modes. On average, the performance improvement is 3.6% with the lower processor speed and is 7.2% with the higher speed, respectively. When $W8$ is excluded, the average performance improvement is 8.5% and 12.4%, respectively.

DTM TDP and Thermal Emergency Levels. Figure 13 shows the normalized running time averaged on all workloads on PE1950 when the thermal design point (TDP) of AMB changes. The thermal emergency levels also change with the AMB TDPs,

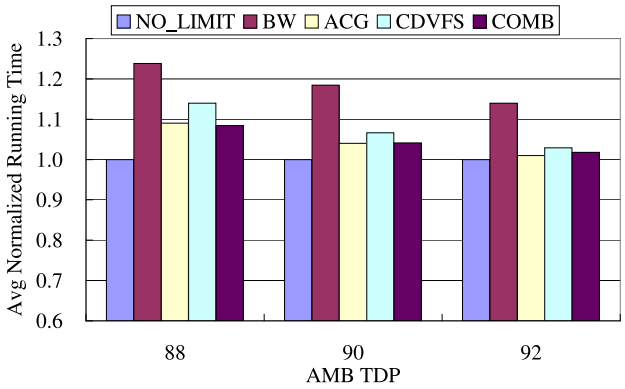


Figure 13: Normalized running time averaged for all workloads on PE1950 with different AMB TDPs.

following the rationales discussed in Section 3. The performance of three AMB TDPs is shown: 88°C, 90°C and 92°C. As expected, the performance loss is reduced with higher TDPs. Compared with that of no-limit, the performance of DTM-BW is degraded by 23.8%, 18.5% and 14.0% with AMB TDPs of 88°C, 90°C and 92°C, respectively. The performance improvement by three policies over DTM-BW is similar under different AMB TDPs. The performance improvement by DTM-ACG is 13.6%, 13.3% and 12.9%, respectively. They are 8.6%, 10.8% and 10.8% by DTM-CDVFS and 14.2%, 13.1% and 12.0% by DTM-COMB, respectively. This result indicates that the three policies may work similarly in systems with different thermal constraints.

Switching Frequency in Linux Scheduling for DTM-ACG. In DTM-ACG, two programs may share a processor core when another core is disabled. The time quantum used in process scheduling is set to 100ms in the kernel by default. Figure 14 compares the normalized running time and number of L2 cache misses averaged for all workloads on PE1950 with different time quantum settings. The running time and number of L2 cache misses are normalized to those with default time quantum for each workload. The results show that the average normalized running time does not have visible changes when the base time quantum is longer than 20ms. When it is set to a value shorter than 20ms, both running time and number L2 cache misses increase steadily. The average running time is increased by 4.2% and 7.2% when the base time quantum is set to 10ms and 5ms, respectively. We find that the major reason

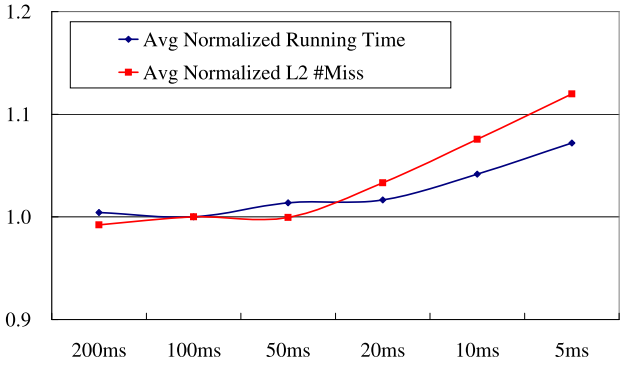


Figure 14: Normalized running time and number of L2 cache misses averaged for all workloads on PE1950 with different switching frequencies.

for the performance degradation is the increase of L2 cache misses. The average number of L2 cache misses is increased by 7.6% and 12.0%, respectively. This indicates that to avoid cache thrashing with DTM-ACG, the default time slice cannot be shorter than 20ms for the processors with 4MB L2 cache used in our experiments.

6. CONCLUSION

We have performed the first study of software dynamic thermal management (DTM) of memory subsystems on multicore systems running Linux OS. It has validated the effectiveness of memory DTM methods in real systems with a new finding on the thermal interaction between the processor and memory in real machines. Future work includes the correction of the previous memory thermal model [15], more evaluation using commercial workloads and varying thermal constraints, and the design of new memory DTM policies using control methods and additional inputs.

Aknowledgement

We appreciate the constructive comments from the anonymous reviewers. This work is supported in part by the National Science Foundation under grants CPA-0541408, CPA-0541366, CSR-0720719 and CSR-0720609.

7. REFERENCES

[1] D. Brooks and M. Martonosi. Dynamic thermal management for high-performance microprocessors. In *Proceedings of the 7th International Symposium on High-Performance Computer Architecture*, 2001.

[2] J. Choi, Y. Kim, and A. Sivasubramaniam. Modeling and managing thermal profiles of rack-mounted servers with thermostat. In *Proceedins of the 13th International Symposium on High-Performance Computer Architecture*, 2007.

[3] J. Donald and M. Martonosi. Techniques for multicore thermal management: Classification and new exploration. In *Proceedings of the 33rd International Symposium on Computer Architecture*, 2006.

[4] B. Ganesh, A. Jaleel, D. Wang, and B. Jacob. Fully-buffered DIMM memory architectures: Understanding mechanisms, overheads and scaling. In *Proceedings of the 13th International Symposium on High Performance Computer Architecture*, 2007.

[5] S. Gurumurthi, A. Sivasubramaniam, and V. K. Natarajan. Disk drive roadmap from the thermal perspective: A case for dynamic thermal management. In *Proceedings of the 32nd International Symposium on Computer Architecture*, 2005.

[6] T. Heath, A. P. Centeno, P. George, L. Ramos, Y. Jaluria, and R. Bianchini. Mercury and Freon: temperature emulation and management for server systems. In *Proceedings of the 12th international conference on Architectural support for programming languages and operating systems*, 2006.

[7] Hewlett-Packed Development Company. *Perfmon project*. `http://www.hpl.hp.com/research/linux/perfmon`.

[8] Intel Corp. Dual-core Intel® Xeon® processor 5000 series. `ftp://download.intel.com/design/Xeon/datashts/31307901.pdf`, 2006.

[9] Intel Corp. Intel® fully buffered DIMM specification addendum. `http://www.intel.com/technology/memory/FBDIMM/spec/Intel_FBD_Spec_Addendum_rev_p9.pdf`, 2006.

[10] C. Isci, A. Buyuktosunoglu, C.-Y. Cher, P. Bose, and M. Martonosi. An analysis of efficient multi-core global power management policies: Maximizing performance for a given power budget. In *Proceedings of the 39th International Symposium on Microarchitecture*, 2006.

[11] C. Isci, G. Contreras, and M. Martonosi. Live, runtime phase monitoring and prediction on real systems with application to dynamic power management. In *Proceedings of the 39th International Symposium on Microarchitecture*, 2006.

[12] J. Iyer, C. L. Hall, J. Shi, and Y. Huang. System memory power and thermal management in platforms built on Intel® Centrino® Duo mobile technology. *Intel Technology Journal*, 10, 2006.

[13] Y. Kim, S. Gurumurthi, and A. Sivasubramaniam. Understanding the performance-temperature interactions in disk I/O of server workloads. In *Proceedings of the 12th International Symposium on High-Performance Computer Architecture*, 2006.

[14] Y. Li, B. Lee, D. Brooks, Z. Hu, and K. Skadron. CMP design space exploration subject to physical constraints. In *Proceedins of the 12th International Symposium on High-Performance Computer Architecture*, 2006.

[15] J. Lin, H. Zheng, Z. Zhu, H. David, and Z. Zhang. Thermal modeling and management of DRAM memory systems. In *Proceedings of the 34th annual International Symposium on Computer Architecture*, 2007.

[16] K. Man. Bensley FB-DIMM performance/thermal management, 2006. Intel Developer Forum.

[17] Micron Technology, Inc. DDR2 SDRAM system-power calculator. `http://www.micron.com/support/designsupport/tools/powercalc/powercalc`.

[18] J. Moore, J. Chase, P. Ranganathan, and R. Sharma. Temperature-aware resource assignment in data centers. In *Proceedins of USENIX*, 2005.

[19] K. Skadron, T. Abdelzaher, and M. R. Stan. Control-theoretic techniques and thermal-RC modeling for accurate and localized dynamic thermal management. In *Proceedings of the 8th International Symposium on High-Performance Computer Architecture*, 2002.

[20] K. Skadron, M. R. Stan, W. Huang, S. Velusamy, K. Sankaranarayanan, and D. Tarjan. Temperature-aware microarchitecture. In *Proceedings of the 30th International Symposium on Computer Architecture*, 2003.