2019 IEEE International Symposium on High Performance Computer Architecture (HPCA)

Understanding the Impact of Socket Density in Density Optimized Servers

Manish Arora*‡, Matt Skach†, Wei Huang‡, Xudong An‡, Jason Mars†, Lingjia Tang† and Dean M. Tullsen*
*University of California, San Diego
†University of Michigan
‡AMD Research, Advanced Micro Devices, Inc.

***Abstract*—The increasing demand for computational power has led to the creation and deployment of large-scale data centers. During the last few years, data centers have seen improvements aimed at increasing computational density – the amount of throughput that can be achieved within the allocated physical footprint. This need to pack more compute in the same physical space has led to density optimized server designs. Density optimized servers push compute density significantly beyond what can be achieved by blade servers by using innovative modular chassis based designs.**

This paper presents a comprehensive analysis of the impact of socket density on intra-server thermals and demonstrates that increased socket density inside the server leads to large temperature variations among sockets due to inter-socket thermal coupling. The paper shows that traditional chip-level and data center-level temperature-aware scheduling techniques do not work well for thermally-coupled sockets. The paper proposes new scheduling techniques that account for the thermals of the socket a task is scheduled on, as well as thermally coupled nearby sockets. The proposed mechanisms provide 2.5% to 6.5% performance improvements across various workloads and as much as 17% over traditional temperature-aware schedulers for computation-heavy workloads.

***Keywords*-Server; Data center; Density Optimized Server; Scheduling**

I. Introduction

The last decade has seen an increase in data center deployments in the enterprise as well as by cloud service providers. With the continued push towards consolidated systems in the enterprise, and the emergence of new applications in domains such as real-time data analytics, industrial IOT and deep learning, this trend is expected to continue [10] [37]. This has resulted in demand for *Density Optimized Servers.*

Density optimized servers consist of a chassis that provides power and cooling. Compute, memory, storage, and connectivity are organized in the form of cartridges. Upgrades to servers can be performed by upgrading individual cartridges. Various combinations of cartridges can be used to create a modular server design where the server is heavily optimized for different workloads such as compute or storage.

Examples of density optimized servers include the HPE Moonshot [15] M700 [12] based systems that are targeted at enterprise virtual desktop infrastructure (VDI) applications. The HPE Moonshot packs forty-five cartridges, each with up to four AMD Opteron^{TM}X2150 [2] sockets in a 4U form factor. Another example is the Cisco UCS M-Series [3] modular servers. The UCS M-Series has eight compute

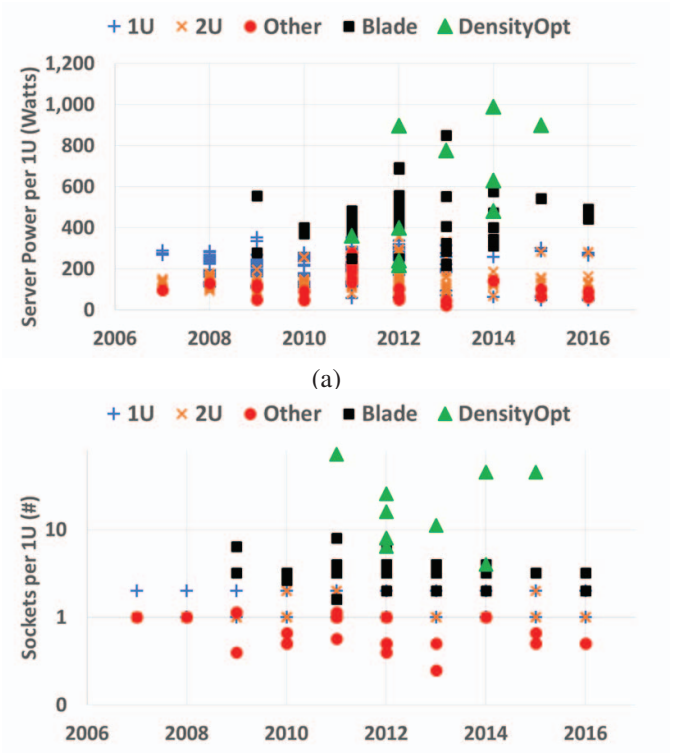


Figure 1: **(a) Power per 1U, and (b) Sockets per 1U for 400 server designs released from 2007-2016.**

modules, each with two Intel Xeon E3 sockets [20] in a 2U form factor. Similarly, Dell has a lineup of density optimized servers with the PowerEdge FX2 [7] chassis-based solutions. More recently, Facebook and QCT have released the Rackgo X Yosemite Valley dense-server design [23] that packs 12 Intel Xeon D-1500 [19] sockets in a 2U form factor.

As server shipments continue to rise, the density optimized server market is growing at a much faster rate than the overall market [16] [17]. According to IDC, in 2015 density optimized servers (also called modular hybrid-blade servers) represented 10% of the server market and they are expected to rise to represent 30% of the server market share by 2020 [8].

The market for density optimized servers is growing rapidly because these designs offer distinct advantages. First, they offer the ability to customize server hardware for the application. Second, these servers optimize for the physical space and hence reduce data center build out costs [5].

2378-203X/19/$31.00 ©2019 IEEE
DOI 10.1109/HPCA.2019.00066

Third, reduced physical footprint enables more efficient (e.g., smaller or cheaper) cooling systems [5]. Fourth, as the physical footprint shrinks, operational, maintenance, and IT administration costs are reduced [10].

Density optimized servers provide these advantages by packing more CPU sockets per unit volume than traditional server designs. Figure 1 shows socket and power density for 400 servers from published results of SPECpower_ssj® 2008 benchmark data [38]. All published data from 2007-2016 was considered except tower servers. Data for density optimized servers was estimated from manufacturer specifications.

Figure 1 shows that 1U servers consume more power per unit volume on average (208 Watt/U) than 2U (147 Watt/U) and "Other" rack server designs (114 Watt/U). This can be attributed to the higher socket density of 1U designs (1.79 Sockets/U) as compared to 2U (1.15 Sockets/U) or "Other" (0.78 Sockets/U). Blade servers exhibit power (421 Watt/U) and socket density (3.47 Sockets/U) that is nearly double that of 1U designs. This is because Blade servers are optimized to pack in more sockets with the use of a shared chassis. Density optimized servers are an evolution of Blade server designs and push the envelope even further. They exhibit power density of 588 Watt/U and socket density of about 25 Sockets/U based on a study of 10 server designs from 4 vendors. This is nearly a 50% increase in power density along with nearly a 6X increase in socket density over Blade server designs.

As density optimized servers gain popularity, it is important to consider the impact of higher power and socket densities on system performance, especially since the individual cartridges share resources. As previous work in balanced system design has demonstrated, performance is not just a matter of increasing the number of cores or size of the machine. Rather, any such increase needs to be accompanied with efficient management techniques to maximize performance [73] [62] [64].

In this paper, we specifically focus on understanding intra-server thermals because of the use of a shared cooling system. With this understanding, we investigate new scheduling techniques that improve performance of density optimized servers. In particular, workload schedulers need to now account for *thermal coupling*. Since many sockets share the cooling system, the placement of a single job not only impacts the temperature profile of the scheduled socket, but potentially every socket downwind of that socket. This asymmetry (upwind sockets impact downwind ones, but not typically vice versa) makes scheduling a challenge.

Thermal coupling is defined as the heat-transfer phenomenon of one heat source affecting the temperature of other heat sources in close physical proximity. Thermal coupling is pervasive in computing systems and previous research has demonstrated the impact of such interactions on cores [72], in 3D stacked systems [66], in CPU-GPU systems [64] and between DRAM DIMMs and CPU sockets in a server system [41]. With increasing socket counts and sharing of the cooling system, sockets within dense servers exhibit strong thermal coupling. This can impact scheduling algorithms.

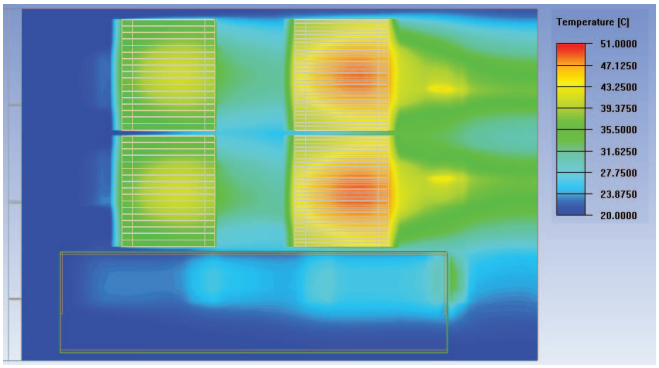


Figure 2: **CFD model of a dense server cartridge.**

This paper makes the following contributions:

(1) Using real-world server configurations this work demonstrates how socket density creates a new type of temperature heterogeneity inside dense servers. This causes significant temperature difference inside the dense server even at reasonable single-socket power and cooling levels.

(2) This work studies temperature-dependent scheduling schemes used at the chip-level as well as data-center system level and demonstrates that existing algorithms are insufficient because they do not take into account the effects of inter-socket thermal coupling caused by directional airflow.

(3) Lastly, the paper proposes a new scheduling scheme that shows gains in performance and energy efficiency. In particular, for computation-heavy workloads, we see as much as a 17% performance gain over a traditional temperature-aware scheduler, and across all load levels, we average 2.5% to 6.5% gains for various workloads.

The rest of the paper is organized as follows. Section II presents an analysis of socket density and temperature heterogeneity in dense servers. Section III describes our evaluation methodology. Section IV describes existing scheduling algorithms, their key shortcomings, and describes our proposed enhancements. Section V presents results, Section VI describes related work, and Section VII concludes the paper.

II. The Impact of Socket Density

Systems such as the HPE Moonshot system can pack in as many as 180 sockets in a 4U chassis. This level of density can create socket to socket interactions that impact performance and efficiency. Using recently proposed density optimized servers as examples, this section analyzes the influence of density on intra-server thermals and application performance.

Figure 2 shows a computational fluid dynamics (CFD) model of a dense server cartridge similar to the HPE Moonshot Proliant M700 [12] constructed using Ansys Icepak. The system has four sockets, arranged in a 2 X 2 configuration. Each socket consumes 15 Watt of power. Air flows horizontally from left to right and passes over

Table I: Recent density optimized systems.

Organization	System	Details	Application Domain	Dimensions	System Organization	Total Sockets	Sockets per 1U	Socket TDP(W)	CPU	Degree of Thermal Coupling
QCT/Facebook	Rackgo X [23]	Open compute server	General purpose	2U	2 tray x 3 blade x 2 socket	12	6	45	Intel Xeon D-1500 [19]	1
AMD	AMD SeaMicro [24]	SM15000e-OP [24]	Scale-out applications	10U	4 row x 16 card x 1 socket	64	6.4	140	AMD Opteron™6300 [1]	1
Cisco	UCS M4308 [3]	M2814 [3]	Scale-out applications	2U	2 row x 2 card x 2 socket	8	4	120	Intel Xeon E5 [20]	1
HP Enterprise	Moonshot [15]	ProLiant M710P [13]	Big data analytics	4U	15 row x 3 cartridge x 1 socket	45	11.25	69	Intel Xeon E3 [20]	2
Dell	Copper [6]	Prototype system [6]	Scale-out applications	3U	12 sled x 4 socket	48	16	15	32-bit ARM® [6]	3
Mitac	Datun project [9]	Prototype system [9]	Scale-out applications	1U	2 row x 4 socket	8	8	50	Applied Micro X-Gene [9]	3
Seamicro	SeaMicro [26]	SM15000-64 [26]	Scale-out applications	10U	4 row x 16 card x 4 socket	256	25.6	8.5	Intel Atom N570 [18]	3
HP Enterprise	Moonshot [15]	ProLiant M350 [11]	Web hosting	4U	15 row x 3 cartridge x 4 socket	180	45	20	Intel Atom C2750 [21]	5
HP Enterprise	Moonshot [15]	ProLiant M700 [12]	Virtual desktop (VDI)	4U	15 row x 3 cartridge x 4 socket	180	45	22	AMD Opteron X2150 [2]	5
HP Enterprise	Moonshot [15]	ProLiant M800 [14]	Digital signal processing	4U	15 row x 3 cartridge x 4 socket	180	45	14	TI Keystone II [27]	5
HP	Redstone [34]	Development server	Scale-out applications	4U	4 tray x 6 row x 3 cartridge x 4 socket	288	72	5	Calxeda EnergyCore [34]	11

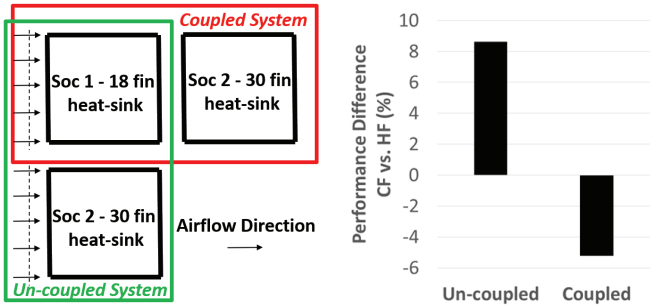


Figure 3: **(a) Organization, and (b) Relative performance of Coolest First (CF) and Hottest First (HF) scheduling for coupled and un-coupled designs.**

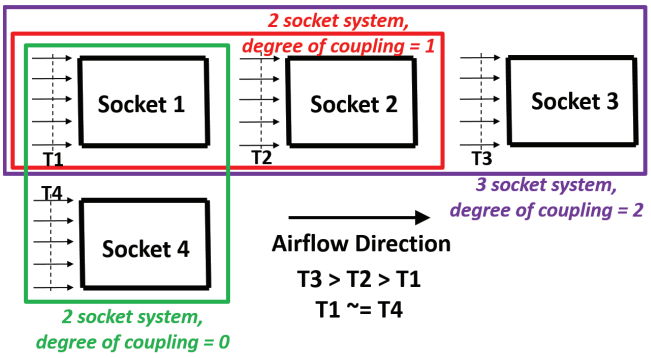


Figure 4: **Thermal coupling in density optimized servers.**

sockets in series. Figure 2 shows that, while cool air flows over the sockets on the left, the sockets on the right receive air that is higher in temperature. The measured average air temperature difference between the left and right sockets was 8C. Consequently, the second set of downstream sockets have a higher ambient temperature and may reduce performance when operating under a temperature limit.

To mitigate chip temperature differences because of inter-socket thermal coupling, the cartridge uses 2 types of heat sinks. The upstream sockets that receive cooler air have 18 fins in the heat sink versus 30 fins for the downstream sockets. The use of distinct heat sinks adds yet another dimension to the thermal heterogeneity in the system, increasing the difficulty of making optimal scheduling decisions. For example, we can choose to place a job on the socket with the coolest ambient air, but that choice heats up at least 1 more downstream socket.

Figure 3 (a) shows a 2-socket system with different heat sinks arranged in a coupled (similar to the cartridge) and un-coupled manner (similar to traditional 2-socket 1U server). Part (b) shows the relative performance when using two scheduling schemes, Coolest First (CF) and Hottest First (HF) at 50% utilization. The CF scheme schedules jobs on the coolest available core and keeps work away from the hotter cores. CF-style scheduling algorithms have been demonstrated to work well as scheduling algorithms [57] [63] [80]. The HF scheme does the exact opposite – it schedules work at the hottest possible idle socket, which would not generally be expected to be a good strategy.

As expected, for an uncoupled system, CF outperforms HF by about 8%. However, for a coupled system, HF outperforms CF by about 5%. This happens because the HF scheme mitigates thermal coupling effects by scheduling less work on the upstream socket. This results in lower air temperature at the downstream socket. The better heat sink at the downstream socket also results in better performance. This simple motivational experiment shows that scheduling policies such as CF that have worked well for chip-level or data-center level scheduling may not work for thermally coupled systems.

A. Socket Organization in Dense Servers

An important factor that differentiates density-optimized servers from traditional designs is socket organization. As demonstrated in the previous section, differences in socket organization (e.g., coupled versus un-coupled) can lead to different performance of scheduling schemes. In this section, we will look at how sockets are organized in current density optimized systems and using an analytical model of heat-transfer, we estimate temperature variations across sockets inside density optimized systems.

Figure 4 illustrates some choices around socket organization in dense systems. As shown in the figure, sockets can be organized in a coupled or un-coupled manner. More than

two sockets can also be organized in a coupled manner to yield a tighter system fit but at a higher *degree of coupling*. We define degree of coupling as the maximum number of sockets that may have a significant thermal interaction caused by sharing of the cooling system.

Figure 4 also demonstrates *socket entry temperature*. We define socket entry temperature as the average temperature of air before it passes over a socket. As shown, systems with higher degree of thermal coupling and shared cooling show progressively higher entry temperatures if all sockets consume power.

Table I shows recently released density optimized server systems and compares them across various dimensions. Density for such systems varies from about 4 Sockets/U to as high as 72 Sockets/U. The systems with higher socket densities tend to use lower power sockets. A study of system organization shows that systems are organized in modular form as rows/trays of cartridges/boards, each with multiple sockets. Power for the sockets used in these systems varies from very low power cores at 5W per socket to as high as 140W per socket. The degree of thermal coupling varies from 1 to as high as 11.

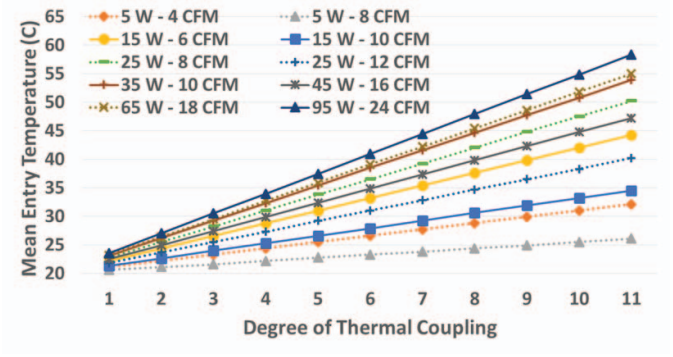
Table I shows that the design space of density optimized systems is fairly large. To consider the impact on inter-socket thermals of various socket organization choices, we construct an analytical model of socket entry temperature for various power, degree of coupling, and airflow levels.

Table II: Airflow requirements for server systems.

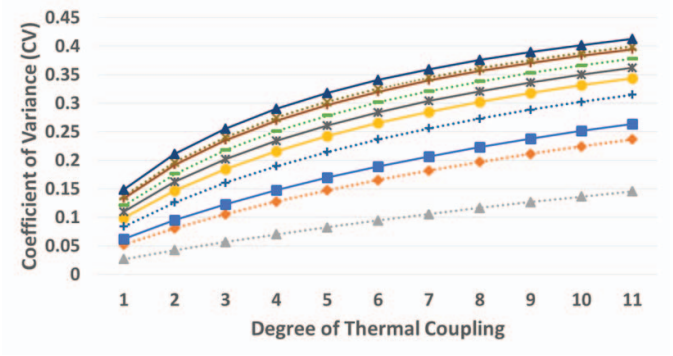
Server Size	Power per 1U (W)	Air-flow (CFM) needs per 1U (DeltaT = 20C)
1U	208	18.30
2U	147	12.94
Other	114	10.03
Blade	421	37.05
DensityOpt	588	51.74

B. Analytical Model of Socket Entry Temperature

In order to see the impact of socket density on intra-server thermals, we build a simple analytical model based on heat transfer theory – this complements our more complex models later in the paper. Electronic systems using forced air cooling rely on the transfer of heat between hot components and cold air being pushed through by fans in order to maintain the temperature. One of the critical limits imposed on server systems is the hot-aisle temperature limit. This limit sets the maximum temperature of air from the server outlet for human comfort. For example, Facebook data center hot aisle temperatures can be as high as 49C with inlets set to about 29C [32]. This means that there must be enough air-flow provisioned to remove heat and maintain an outlet-inlet temperature difference of 20C. ASHRAE TC 9.9 guidelines [30] also mention a typical server temperature rise of 20C from inlet to outlet.



(a)



(b)

Figure 5: **(a) Mean socket entry temperature, and (b) Socket entry temperature coefficient of variance, a measure of the variance in heat across the coupled sockets [4].**

Table II shows the air-flow requirements in order to maintain a 20C temperature difference between the inlet and outlet for various server systems. Air flow in the table is measured in cubic feet per minute (CFM) and power numbers are average server power levels as previously discussed in section I. CFM levels were calculated using the standardized total cooling requirements formulation of the first law of thermodynamics [25]. Table II shows that to maintain a temperature difference of 20C between the inlet and the outlet, fans need to supply between 18.3CFM to 51.74CFM of air per 1U for different server power levels. High-end server fans, such as Activecool fans [29], can meet such airflow requirements at reasonable power levels.

While the total CFM requirements are determined by the outlet temperature limits, the temperature inside the server is dependent on the socket power, degree of coupling, and the airflow on each socket. We use the values in Table II to calculate reasonable bounds on per-socket air flow. Next, we use socket power and degree of coupling combinations, with the standardized total cooling requirements formulation of the first law of thermodynamics [25] to derive the temperature of air arriving at points within the server. Figure 5 shows the results of our analysis.

Figure 5(a) demonstrates that as the degree of coupling increases, the mean socket entry temperatures also increases. As expected, the mean entry temperature for high-powered

sockets is high, but even low-powered sockets can have high mean entry temperature at different airflow values. For example, a 15 Watt part with 6CFM of airflow can have about a 10C mean entry temperature difference for a system with degree of coupling 5, as compared to a system with degree of coupling 1. Figure 5(b) shows that not only is the mean socket entry temperature high, but inter-socket variations are also high and increase even further as the degree of coupling increases. The data demonstrates that socket organization can play a major role in intra-server thermals for systems with high degree of thermal coupling.

As table I shows, density optimized systems vary in organization, processor choices and application domains. In order to study optimization opportunities, we pick a single representative system used for its intended application domain. We pick a system similar in design to the HPE Moonshot Proliant M700 system as our system under test (SUT) for this deeper analysis. A key target application market for such systems is Virtual Desktop Infrastructure (VDI) intended for enterprise desktop consolidation [12]. With 180 sockets packed in 4U space, the SUT is highly representative of density optimized systems and is intended for a popular enterprise data-center use-case [10]. Next, we discuss infrastructure used to model our target system.

III. Methodology

This section describes details of our modeling infrastructure. Our SUT consists of 180 sockets and targets VDI applications. Since we are modeling a large-scale system consisting of many sockets, and running workloads over a period of minutes, we develop a methodology that enables us to accurately simulate the behavior of such systems in reasonable simulation time.

A. Workloads

For the purposes of this study we focus on VDI, where the servers are running desktop applications in support of thin clients or terminals.

To study desktop application behavior, we used PCMark® 7 [31]. PCMark is a popular consumer application benchmarking suite, commonly used in industry to characterize desktop performance. PCMark consists of more than 25 applications characterized into domains such as computation, storage, graphics, entertainment, creativity, and productivity. Out of these, we omit applications that are not relevant to enterprise VDI (e.g., gaming) for a total of 19 PCMark 7 applications. To easily categorize benchmarks, we divide these remaining applications into 3 sets: Computation intensive (Computation), Storage intensive (Storage), and General Purpose (GP) benchmarks.

Due to the unavailability of timing-accurate public domain simulation tools that can execute typical desktop applications on Windows®, we use a trace based simulation methodology similar to that used in previous research using PCMark 7 [39]. We capture hardware traces of various PCMark benchmark

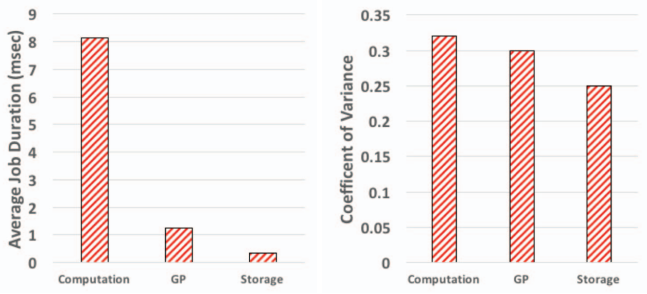


Figure 6: **(a) Average job duration, and (b) Coefficient of variance of job durations within benchmark sets.**

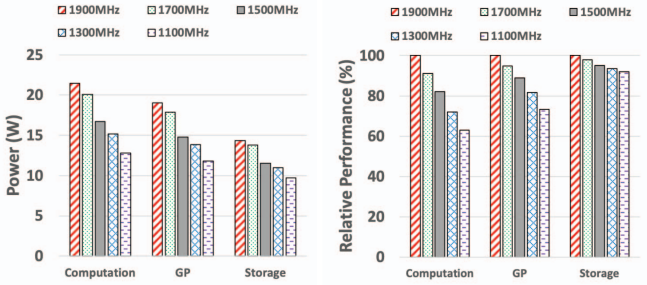


Figure 7: **(a) Workload power (90C) with varying frequencies. (b) Relative workload performance versus performance at 1900MHz.**

runs using the Windows Xperf tool [28] at various processor frequency states. Xperf captures idle and active transitions of the socket for the workload at a fine granularity. Using this information, we create a job arrival model for various PCMark benchmarks. We vary the job inter-arrival duration to simulate different loads on the system.

Figure 6(a) shows average job durations across all benchmarks for the three benchmark sets. The average job durations were found to be on the order of a few msec. The maximum job durations within each benchmark set were found to be almost 2 orders of magnitude higher. This is in line with previously published analysis of PCMark 7 [39]. Figure 6(b) shows the coefficient of variance across average job durations within the different benchmarks of each set. The coefficient of variance ranges between 0.25 to 0.33. This shows that benchmarks within each set exhibit similar job durations on average and hence we choose to study benchmarks grouped in sets, rather than studying benchmarks individually.

We measured power and performance in hardware at various frequency states to build a complete socket-level workload model. As power is influenced by temperature, we measure both temperature and power together, across varied loads. Using the measured power value, temperature value, and by estimating leakage to be 30% of TDP at the temperature limit (90C), we calculate power at different frequencies and chip temperatures. Figure 7 shows the power and performance levels for the different workloads.

The AMD Opteron X2150 used in the system has a TDP of 22 Watts and runs from 1900MHz to 1100MHz [2]. We see that the Computation workload uses the most power and Storage uses the least (18 Watt versus 10.5 Watt) at

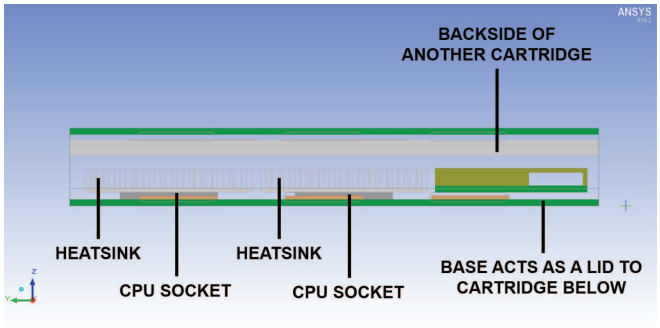


Figure 8: **Side view of modeled cartridge.**

the highest frequency of operation. As frequency decreases, power decreases but more so for Computation than Storage. As expected, the Computation workload is also the most frequency sensitive with performance dropping about 35% for a 800MHz reduction in frequency. Storage is the least frequency sensitive workload. The General Purpose workload exhibits intermediate levels of both power and sensitivity to processor frequency.

B. Socket Ambient Temperature Model

Socket ambient temperature is a function of the physical design, power consumption of sockets, heat sink design and airflow inside the server box. Using available data for the ActiveCool [29] fans, physical dimensions published for the M700 server cartridge [12], and socket power model, we used an Ansys Icepak based modeling infrastructure to construct a model of our SUT. This computationally intensive model tracks heat sources, physical artifacts, fans, and turbulent airflow through the system. This commercial CFD based modeling infrastructure and methodology has been validated on real server designs to be within 1C of actual temperature within the server box [71]. This modeling infrastructure yields socket ambient air flow levels and socket temperature based on the server physical design and different socket power consumptions.

Figure 8 shows the side view of a single cartridge. The figure shows the model for a single cartridge as well as the second row acting as a lid on top of the first cartridge with its backside facing the first cartridge. The SUT has 15 rows of cartridges spanning the width of the server, stacked on top of each other in this side view. We only examine thermal coupling along the direction of airflow in this work. There is some thermal coupling between cartridges in the z direction across the width of the server; however, our CFD modeling confirms that these are small effects compared to the thermal coupling we model.

C. Chip Peak Temperature Model

To simulate system behavior with reasonable accuracy, it is important to reliably estimate peak chip temperature for a socket with known ambient temperature and chip power consumption. Architecture research has traditionally used compact thermal models like Hotspot [75] to estimate chip

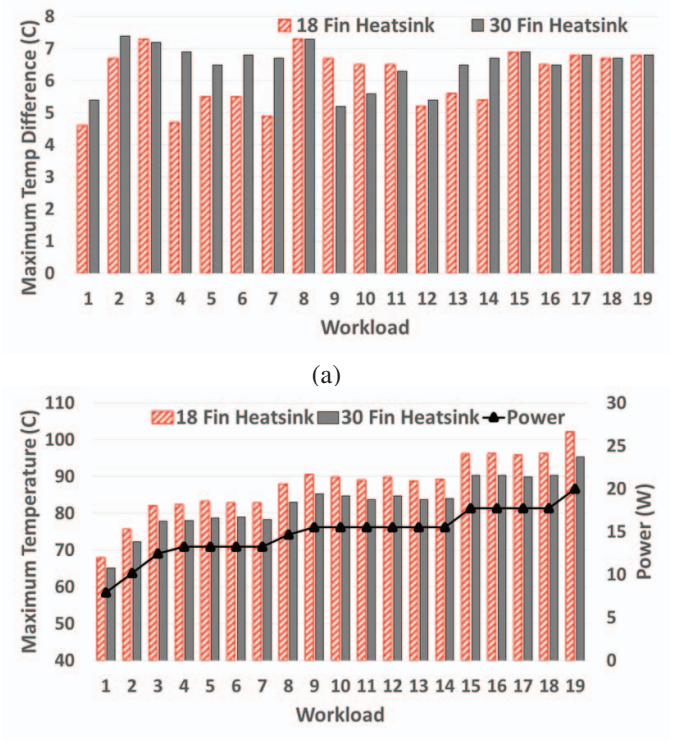


Figure 9: **(a) Temperature differences between hottest and coolest spots, and (b) Max temperature versus power.**

thermals as on-chip temperature differences between cold and hot spots can exceed as much as 50C [65] for large sized dies. However, recent research has also shown that socket level thermals in server systems can have time constants of the order of tens of seconds [40] [64]. Hence, traditional detailed modeling approaches becomes prohibitively expensive when modeling a dense server system with many sockets and running workloads for periods of seconds to minutes.

In order to explore the development of simpler thermal models for peak temperature estimation, we studied on-chip temperature differences for the AMD Opteron X2150 [2]. Figure 9(a) shows data for temperature differences between the hottest and coldest spots on the die for 19 PCMark 7 benchmarks. The data was collected via a proprietary HotSpot like model that has been validated with thermal camera measurements.

The data shows that different heat sinks do not have a major impact on chip lateral temperature differences, although, in general, the 18 fin heatsink had lower chip lateral temperature differences than the 30 fin heatsink. Interestingly, we see that the temperature differences on die are fairly low for the X2150 and range between 4C - 7C. We attribute this to the small size of the die at about $100mm^2$ [35], about $3.5\times$ – $6\times$ smaller than server processor dies [22].

Figure 9(b) shows the maximum chip temperature and power for various PCMark 7 benchmarks using the validated temperature model. The data shows that the 30 fin heatsink

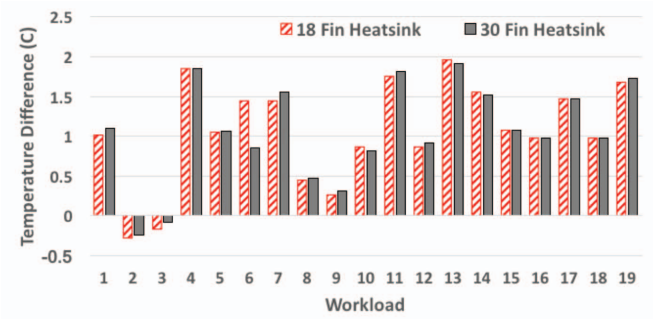


Figure 10: **Proposed max temperature model validation.**

provides better thermals than the 18 fin heatsink by about 6C - 7C for high power cases and 3C - 4C for lower power cases. Also, the peak temperature is well correlated with total power. Using data and insights from the validated temperature model, we develop a simplified peak temperature prediction model for the X2150. We approximate the core peak temperature as:

$$T_{Peak} = T_{Amb} + Power \times (R_{Int} + R_{Ext}) + \theta(Power, Sink) \quad (1)$$

where T_{Peak} is the peak chip temperature, T_{Amb} is the socket ambient temperature, R_{Int} and R_{Ext} are the chip internal and heatsink external thermal resistance, and $\theta(Power, Sink)$ is a linear function that is derived empirically. The model considers the thermal resistance from the die to the external environment but ignores any lateral thermal resistance. The developed model is similar conceptually to the simplified chip temperature model proposed in [42], but additionally models the path from the heatsink to the ambient.

Figure 10 shows the differences in temperature between our proposed simplified model and the validated proprietary thermal model. The figure shows that our proposed simplified model estimates temperature within 2C of the validated model. This holds valid irrespective of the heatsink size.

D. Overall Model

In this section, we will describe the overall model. Table III provides values of different parameters used in our model along with information as to how they were derived.

In our simulation model, jobs for various benchmarks arrive as per a probabilistic job distribution model created to match the Xperf measurements previously described. Once the job arrives, it is put into a job queue, which is used by a centralized job controller to allocate jobs. The scheduler checks for job arrivals every 1 *usec*. If there is a job to be scheduled and at least one idle socket available, the scheduler implements policies based on current and historical temperature, physical location, job power, and other parameters.

The scheduler works by first making a list of all sockets that are idle. It then makes a decision to allocate the job from the queue, based on the scheduling policy, on to one of the free sockets. If a free socket is not available, the scheduler does not make any allocation decision and waits to make

Table III: Overall simulation model parameters.

Parameter	Value	Methodology
Job length	Variable	Captured using Xperf [28]
Job arrival time	Variable with load	Captured using Xperf [28]
Job power	Variable	Measured in hardware
Job performance	Variable with PStates	Measured in hardware
Frequency	1900MHz - 1100MHz	Product data sheet [2]
Temperature limit	95C	Typical
Frequency change interval	1msec	From [64]
Power management	Highest frequency allowed under 95C	Typical for responsive systems [64] [67]
On-Chip thermal time constant	5msec	Typical
Socket thermal time constant	30 seconds	From [67]
Server inlet temperature	18C	Typical
Server total airflow	400CFM	Calculated from fan data [29]
Airflow at sockets	6.35CFM	Estimated using Ansys Icepak model
Ambient temperature	Variable	Estimated using Ansys Icepak model
R_{Int}	0.205 Celsius/Watt	Calculated using Hotspot [75]
R_{Ext} 18-fin	1.578 Celsius/Watt	Calculated using Hotspot [75]
R_{Ext} 30-fin	1.056 Celsius/Watt	Calculated using Hotspot [75]
$\theta(Power, 18-fin)$	4.41 - Power x 0.0896	Modeled
$\theta(Power, 30-fin)$	4.45 - Power x 0.0916	Modeled
Simulation time	30 minutes of server time	Simulate at least 10M jobs

decisions until a socket becomes available. In this study, we do not implement unbalanced schedulers that choose to schedule jobs to busy sockets even though there is an empty socket available.

Once the allocation of jobs is complete, the sockets execute work every time-step until the next job arrives. Throughput of every time interval and power are estimated based on the current frequency. Estimated power values are used to calculate the current peak temperature of each socket.

The peak temperature value is used to make frequency change decisions. Frequencies vary from 1.9GHz to 1.1GHz in 200MHz steps. The higher two frequencies are boost states that are used opportunistically to improve performance where thermal headroom is available. A fully loaded socket under reasonable ambient temperatures is expected to only be able to sustain the highest non-boosted frequency (1500MHz) [36]. We implement a power management policy that emphasizes responsiveness and runs jobs at the highest possible frequency within the temperature limit. This is a commonly used policy in consumer systems that emphasize responsiveness [64]. The power manager runs every 1*msec*.

The power manager also implements power gating for idle sockets. At every run of the power manager, it checks for sockets that were idle completely during the last 1*msec* and power gates them instantaneously. We assume that power gated sockets still consume 10% of TDP power. Once a job is complete, its performance is recorded by calculating the time it took to finish the job. Overall performance for each job allocation scheme is calculated by finding the cumulative time the system took to complete all jobs.

IV. Scheduling Techniques

The uni-directional flow of air from one end of the server to the other is the primary factor leading to inter-socket

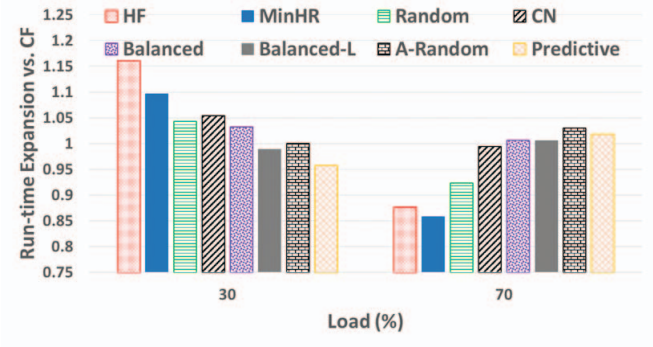


Figure 11: **Average runtime expansion vs. CF for existing thermal aware scheduling mechanisms at 30% and 70% load for the Computation workload (lower is better).**

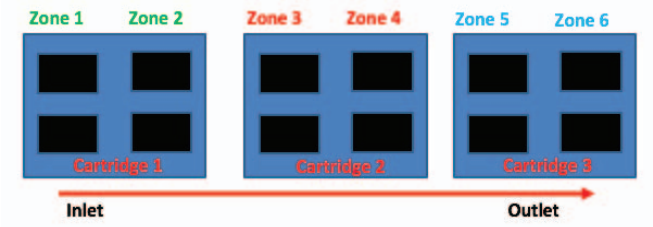


Figure 12: **Organization of zones in the SUT. Each row consists of 3 cartridges (12 sockets) and 6 zones. There are a total of 15 rows in the system.**

temperature variance. In such systems, hot air that has already absorbed heat from one socket ends up flowing over a downstream socket causing different sockets to thermally couple.

Thermal coupling is fairly well-studied in existing computing systems. Thermal coupling exists among cores in a multi-core system [59], between the CPU and the GPU in a heterogeneous multi-core system [64], between cores across layers in a 3D stacked processor [66], and between sockets and DRAM in server systems [41]. At the data-center level, thermal coupling occurs vertically among servers in a rack [50] and across regions of a data center [63] [69].

However, the problem of thermal coupling for density optimized servers has different properties. There are three main differences. First, thermal coupling in this context is primarily uni-directional because of the air-flow direction. Second, thermal coupling is unavoidable, at least amongst certain sockets, due to their physical proximity. Last, as discussed before, the degree of coupling is fairly severe.

Because of these differences, traditional scheduling techniques that have been used to manage thermal behavior may not be sufficient. The following section describes state-of-the-art work in scheduling techniques. Following, we analyze existing techniques in context of our SUT. Last, we propose a new scheduler that outperforms existing schemes.

A. *Existing Scheduling Techniques*

First, we examine existing scheduling techniques used to mitigate thermal effects at the data center level.

Coolest First (CF). The CF policy [63] [76] [80] assigns jobs to the coldest compute elements in the data-center in order to add heat to cool areas and remove it from warm areas. We use instantaneous socket temperatures within our dense server as a metric to implement this scheme. This temperature is maintained within our scheduler and updated every time step. We also implement the Hottest First (HF) scheme. HF is the exactly opposite policy to CF and schedules more work on the warmer areas of the system, amongst the idle sockets.

Minimize Heat Recirculation (MinHR). The MinHR [63] policy assigns jobs so as to minimize the impact of heat recirculation in the data center. The implementation of this scheme involves running (offline) reference workloads at different servers and measuring temperature throughout the data center to calculate heat recirculation factors. Others have proposed better heat transfer modeling techniques [77] [78] to implement similar policies. We implement this scheme by assigning power to sockets and measuring thermal coupling throughout the server to make a fixed heat-transfer map of the dense server. At run-time, the scheduler uses this map and assigns jobs to the idle socket that causes the least thermal coupling in the system.

Random. The Random scheduling policy [63] [76] assigns jobs randomly across idle components in order to approach the behavior of uniform power consumption and thermals.

Next, we examine thermal scheduling strategies previously proposed at the chip level.

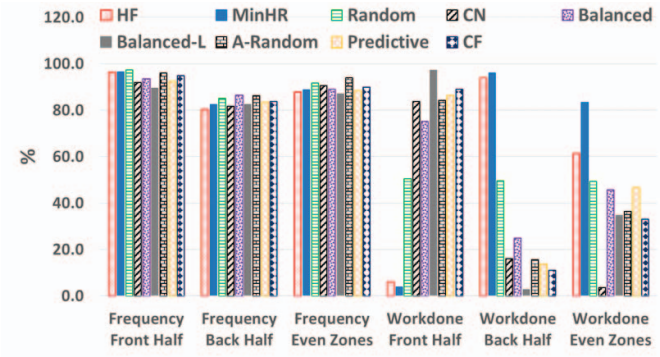
Coolest Neighbors (CN). The CN [54] policy is also a variant of the CF policy. It considers the temperature of each component as well as its neighbors' temperatures to account for on-chip lateral heat-transfer. It assigns jobs to locations that have the coolest neighbors.

Balanced. The Balanced scheduling policy reduces temperature variance by scheduling work to maintain a uniform temperature profile at compute elements [54] [55]. The policy works by scheduling work away from hot spot locations. We implement this policy by scheduling work furthest away from the hottest point in the server.

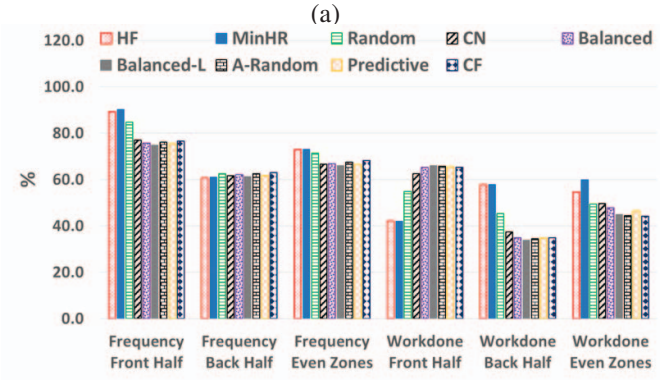
Balanced Locations (Balanced-L). The Balanced-L [55] scheduling policy assigns work to locations that are expected to be the coolest based on their location (e.g., cores on the edges). We implement this policy by giving preference to sockets that are closest to the air-inlets.

Adaptive-Random (A-Random). The A-Random [54] policy is a variant of the CF policy and considers temperature history along with the current temperature. Amongst the components with lowest temperatures, the policy chooses randomly from the ones with lowest historical temperature in order to weed out locations that are consistently hot.

Predictive. The Predictive [81] [43] scheduling policy first calculates the future temperature of a socket if the job were to be scheduled on it. Based on the temperature it predicts the frequency at which the socket can run the job and picks the location that can run the job the fastest.



(a)



(b)

Figure 13: **Average frequency, work performed in front half, back half and even zones of the SUT for various schemes at (a) 30% load, and (b) 70% load.**

B. Analysis

Figure 11 shows a performance comparison across all existing schemes for the Computation workload at two different load levels. Computation is the highest powered workload and also the most sensitive to frequency changes. We measure average run-time expansion across all job completions and baseline the results versus the CF scheme.

We see that at low load, the CF scheme works fairly well and most of the other schemes have similar or worse performance than CF. Only the Predictive scheme is able to significantly improve performance. However, at the higher load, different schemes produce different results. HF and MinHR are the worst performing schemes at the lower load level, but at the higher load MinHR and HF become the best schemes, while Predictive loses its advantages.

Before we explain the reasons for these results, we will define zones in relation to the SUT. As shown in Figure 12, the SUT has three cartridges in series, each like the cartridge in Figure 2. We divide the SUT into 6 zones with each cartridge consisting of two thermally coupled zones. We label the zones 1 - 6. The odd zones have a 18-fin heat sink and the even zones have a 30-fin heat sink. Air flows through the system from zone 1 to zone 6. This creates asymmetric thermal coupling, not just because of the different heat sinks, but because sockets within the same cartridge are only 1.6 inches apart, while adjacent sockets between cartridges (e.g., zones 2 and 3) are about 3 inches apart.

Figure 13 present insights into the behavior of existing schemes for the SUT. It shows two metrics for three different locations of the server. *Frequency* is the average frequency at which the sockets operate relative to the highest possible frequency of operation (1900MHz). *Workdone* is the relative amount of work performed within a specific location as a proportion of overall work. Both these metrics are calculated for the front zones 1-3, back half zones 4-6, and for even zones with the better heat sink.

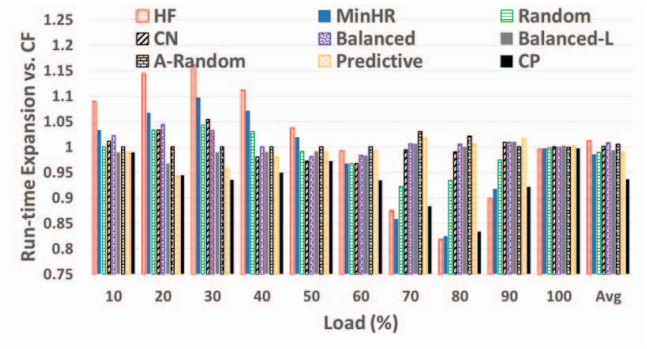
At 30% load, less than half of the system is sufficient to complete all of the work. As seen in Figure 13 (a), all schemes except Random, HF and MinHR allocate most of the work on the front half of the server and are able to sustain high frequency of operation including being in boost states for a considerable amount of time. In this case, using the back half of the server hardly provides any advantage – we can clump jobs near the inlet and leave the downwind sockets idle, minimizing the effect of coupling. The result is that Random, HF, and MinHR, which do not do that, suffer.

CN schedules jobs primarily at zones 1, 3 and 5 in order to choose sockets that are cool as well as have cool neighbors and hence ends up with lower performance versus the other schemes that schedule at the front part of the server. Predictive is the best scheme at 30% load as it is able to choose the fastest socket to run the current job dynamically. As Figure 13 (a) shows, Predictive performs almost 80% of its work in the front half of the server but almost 50% of the work on even zones. Since there is only one even zone in the front half of the server, Predictive is performing most of its work on zone 2, an even zone with the better heat sink. Hence by combining the best from both scenarios (choosing the front part of the server and also the part of the server with the best heat sink), Predictive can outperform all other schemes.

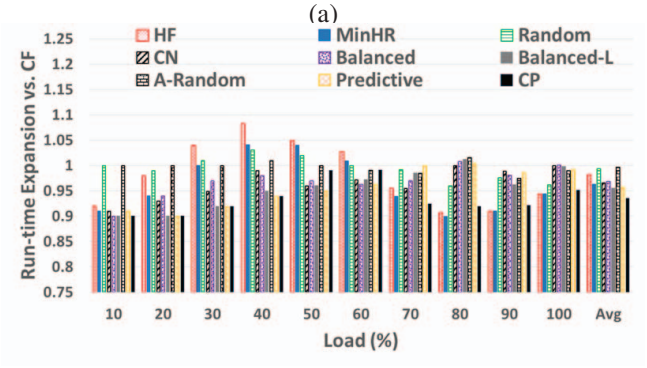
At higher loads, however, we can no longer assume downwind sockets will be idle. At 70% load, we are using the back half more heavily, for most schemes, and the frequency of the back half is more impacted. Schemes like HF and MinHR perform more work at the back of the server and also end up performing more work at even zones, as the back of the server has two out of the three even zones.

An added advantage of packing more jobs in the back is that the front sockets can run at higher frequency as they have less work and hence can sustain boost longer for whatever work they perform. As a result, at 70% load, schemes like MinHR and HF outperform schemes that over-schedule in the front part of the server. Predictive offers no advantage as it schedules work to the coolest areas without regard for thermal coupling and ends up heating both the front and back of the server. In-fact, Random performs better than most other schemes (except MinHR and HF) as it distributes more jobs towards the back of the server than front loading schemes.

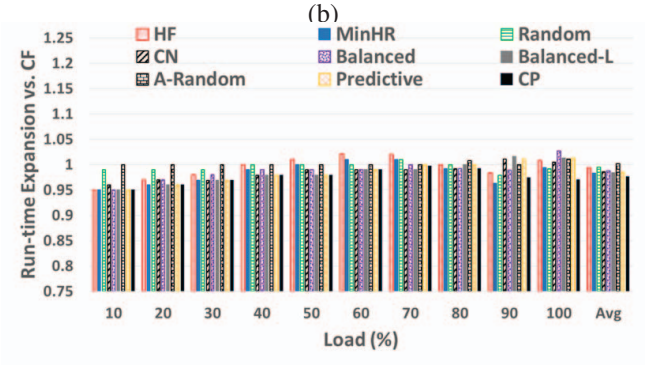
This analysis shows all existing scheduling policies have considerable drawbacks. CN, Random, MinHR and HF leave



(a)



(b)



(c)

Figure 14: **Performance versus CF baseline for various schemes at various load values for different benchmarks (a) Computation, (b) GP, and (c) Storage.**

performance on the table at lower loads by not running at the fastest locations. At the higher load levels, CF, Balanced-L, A-Random, and Predictive do not account for thermal coupling or the heterogeneous structure of heat sinks in the server. In the following section, we will propose and evaluate a new scheme that combines the best features of existing schemes to provide better performance.

C. Proposed Scheme

As seen from the analysis in the previous section, a good scheduling strategy for dense servers balances two factors. First, it should consider the speed at which a job can run at different locations in the server by accounting for ambient temperatures and heat sink properties. Second, it should consider how scheduling at a particular location degrades performance of other downwind locations. The performance degradation of other sockets should be balanced with running the job faster at the socket where the job is to be scheduled. That is, the scheduler should consider the system holistically in making decisions and not just optimize point by point.

Based on these factors, we propose a new scheduling policy named **CouplingPredictor (CP)**. The CP algorithm extends the Predictive [81] [43] algorithm by taking into account thermal coupling effects. CP predicts the performance of not only the socket where the job is scheduled but also the performance of all other sockets that are downstream to this socket. It chooses the socket to schedule that results in overall benefit. Consider two cases, first, a socket that can run the job at 1700MHz but slows down two other downstream sockets total by 300 MHz, and second, a socket that can run the current job at 1600MHz but does not slow down any other sockets. Given such a choice, CP picks the second socket to schedule the job on since it results in overall benefit.

Mechanics. At every time interval, the scheduler checks if any jobs have arrived and scheduling decisions are pending. If there are jobs that need scheduling, the scheduler first picks a row of cartridges with idle sockets at random and then evaluates candidates within that row.

Within the selected row of cartridges, the scheduler first finds a list of idle sockets. Next, for each idle socket, it assumes that the job is scheduled on to it and estimates an initial chip temperature using equation 1. It then updates power by compensating for temperature dependent leakage and predicts final chip temperature, again using equation 1. The scheduler then estimates the highest frequency of operation that keeps the estimated chip temperature less than the temperature limit and saves this value. In addition, based on the ambient temperature model of the system, it uses a table lookup to estimate downwind socket ambient temperature. Then again using equation 1 and assuming the downwind sockets continue to run the same jobs they are running, it predicts frequencies of each downwind socket.

Note that we have sought to keep the scheduler very simple. We use a simple linear coupling model, rather than the complex models we use to evaluate this research. We do not account for varying application slowdowns with respect to frequency, etc. Thus, the gains we show here could be improved with more accurate accounting of these effects, but at a cost of a more computationally expensive scheduler.

V. Results

In this section we will evaluate the proposed CP scheme against other schedulers. Figure 14 shows performance results at different load levels for all of our benchmarks.

As previously discussed, Predictive performed the best amongst existing schemes at low load values. For load values in the 10% – 30% range, CP outperforms or matches Predictive across all benchmarks. For Computation, at 30% load, CP is better than Predictive by about 2%. As compared to other existing schemes, CP provides a 5% – 15% gain over MinHR and about 3% – 7% over other schemes. As

load increases to the 50% – 60% range, Predictive loses its advantage and at higher loads CP is consistently better for the Computation workload.

At low load values, Storage and GP benchmarks both show good performance across schemes versus CF (note that these graphs are normalized to CF – so CF is represented by the 1.0 line). CF packs almost all the work in the first zone. This may lead to throttling of the first zone while scheduling at better heat sinks downwind proves to be advantageous for other schemes. This benefit disappears as load and consequently heat in the system increases. As the load approaches 30% – 40% for the GP workload, we begin to see similar behavior as we saw for 10% – 20% load in the Computation workload. Storage is insensitive to frequency, and hence we see rather muted behavior across all schemes for Storage, except in a few cases such as 10% load.

The GP workload shows the most gain at low loads for the CP scheme. GP has lower benchmark power and almost comparable frequency sensitivity as compared to the Computation workload. Hence it exhibits slightly more opportunity to optimize at lower loads where it sees less throttling than Computation. Both Predictive and CP capitalize on GP and provide gains of about 8% on average versus CF at low loads. Overall, at low loads, CP out performs CF by 3% to 8% and matches the performance of Predictive.

Mid load values (40% load – 60% load) exhibit some interesting behavior. At these load values CP has to continuously choose between optimizing for a single socket versus accounting for thermal coupling. CP is able to make the right decisions in almost all cases except for 50% and 60% load in the GP benchmark where Predictive is about 2% better.

At high load values (70% load to 100% load), HF and MinHR perform well. As seen from Figure 14, at high load levels, the CP scheme is able to match the performance of the HF and MinHR schemes in most cases. This is because CP accounts for the effects of job scheduling on downstream thermally coupled sockets. Computation workload has the highest power, most throttling at high loads, and highest frequency sensitivity. Hence, it presents the highest opportunity to optimize. The CP scheme outperforms CF by 8.5% for Computation on average across high loads and by 6.5% across all load levels. Benefits can be as high as 17% (Computation 80% load). CP also improves the performance of GP by 6% over CF. Storage again sees muted gain of about 2.5%. CP performs significantly better than Predictive at high load values with gains between 2% to 9% across different workloads and as high as 17% (Computation 80% load).

While HF and MinHR exhibit poor performance at low loads for Computation, their relative behavior versus CP improves for GP and Storage workloads. CP continues to beat HF and MinHR but with decreasing margins. At high load values, the relative differences between MinHR and CP reduce where power and frequency sensitivity drops. However, CP is able to compete or beat other schemes that improve performance over certain localized load ranges. Such adaptive and load agnostic behavior is important for server systems where system load can change constantly based on user demand.

If we are to consider averaged performance across all load values, CP out performs all other schemes by at least 5.5% for Computation, 3% for GP and 1.5% for Storage. CP outperforms CF by 6.5% for Computation and GP and about 2.5% for Storage. If we consider individual load values, CP may outperform CF as much as 17% for Computation, 10% for GP and 5% for Storage. We observe that no existing scheme provide consistent performance across all load levels. Existing work tends to optimize at single points such as socket level frequency, or only minimizing heat recirculation which under-performs at certain loads in dense servers. The proposed CP algorithm not only improves performance but also provides robust performance.

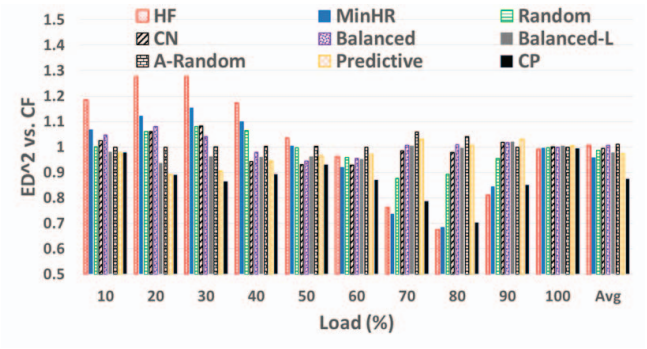
CP works better than existing schemes because it considers inter-socket thermal coupling and carefully weighs scheduling effects on other sockets. Also, decisions are made by evaluating the potential for throttling at each job arrival, allowing it to make decisions at a finer granularity than just considering load as a proxy for thermals.

Figure 15 shows normalized ED^2 product values across different loads and schemes for all of our workloads. For Computation, the ED^2 product drops to as low as $0.7\times$ at 80% load. For GP and Storage, the ED^2 product drops to as low as $0.8\times$ and $0.85\times$ respectively. In general, the ED^2 product of CP tracks that of Predictive at low load values and MinHR at high load values. These results show that CP also matches the energy behavior of Predictive and MinHR at different load values. The CP scheme buys us the performance of these schemes but imposes no extra energy penalties.

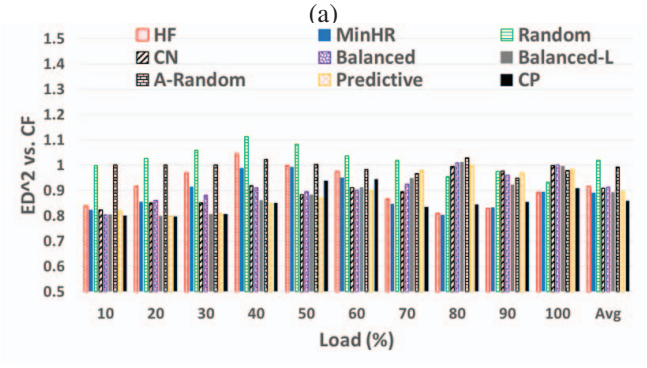
VI. Related Work

There have been a large number of studies examining thermal mitigation in processors [60] [56] [74] [57] [66] [82], servers [79] [50] [40] [42] and in the data center [44] [69] [63] [76]. Thermal mitigation could be achieved via (1) power and thermally efficient system design including micro architectural techniques [48] [72] [66] [61]; (2) power management for energy efficiency [70] [45] [46] [39]; (3) use of efficient packaging and cooling techniques including management of cooling systems [51] [47] [71]; and (4) runtime thermal management including scheduling methods. Our work falls in this last category of dynamic thermal management at runtime (DTM) techniques. Prior DTM research can be further classified in to four categories.

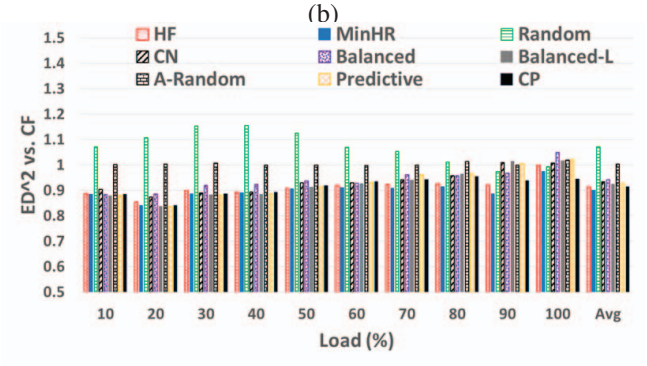
Voltage and Frequency Scaling. The goal of dynamic voltage and frequency scaling techniques is to control processor overheating by keeping the temperature below a critical threshold [33] [67]. Modern processors implement such techniques at finer granularities with the use of temperature estimation or sensor implementations (several thermal entities



(a)



(b)



(c)

Figure 15: **ED2 versus CF baseline for various schemes at various load values for different benchmarks (a) Computation, (b) GP, and (c) Storage.**

per chip [64] or per core DVFS proposals [56]). Research in this area is orthogonal to our work which focuses on job scheduling across sockets in a server and any improved DVFS technique can be used in conjunction with our proposal.

Resource Throttling. Prior research involves controlling the behavior of the processor when it approaches a critical temperature limit. Proposals include global clock gating by Brooks *et al*. [49], local feedback controlled fetch-toggling by Skadron *et al*. [72] and decode-throttling by Sanchez *et al.* [68]. These techniques can be implemented with any scheduler as they manage thermals within a single socket.

Workload Profiling and Migration. Srinivasan *et al.* [74] propose off-line workload analysis techniques to decide processor operating frequencies in order to mitigate thermals. Others propose techniques to migrate work reactively [58] [54] (after reaching a temperature threshold), proactively [52] (before reaching a limit) or predictive [81] [53] (by estimating future temperature). Migration is analogous to scheduling and may be useful when job durations are long. Our scheduling strategy can just as easily be used to choose sockets for workload migration in suitable systems, or even identify when migration would be profitable.

Scheduling and Dynamic Thread Assignment. A variety of scheduling techniques have been proposed in the past including techniques that make decisions based on current temperature [63] [80], history [54], randomized [76], prevent heat re-circulation [63] and predictive temperature estimation [81] [43]. We compare against these schemes and demonstrate better performance across various load levels. Heat-and-Run [57] proposes the loading of cores as much as possible by scheduling threads that use orthogonal resources on to an SMT system. These techniques can be used in conjunction with our research to perform core level scheduling.

VII. Conclusion

This work provides a comprehensive analysis of intra-server thermals for emerging density optimized systems. It shows that existing scheduling algorithms perform sub-optimally across the spectrum of load levels as they do not account for inter-socket thermal coupling. We demonstrate new scheduling techniques that account for heat transfer amongst sockets and resulting thermal throttling. The proposed mechanisms provide 2.5% – 6.5% performance improvements across various workloads and up to 17% over traditional temperature-aware schedulers for computation-heavy workloads.

Trademark Attribution

References

[1] AMD Opteron 6300 Series Processors. http://www.amd.com/en-us/products/server/opteron/6000/6300.

[2] AMD Opteron x2150 Processor. http://www.amd.com/en-us/products/server/opteron-x/x2150.

[3] Cisco UCS M-Series Modular Servers. http://www.cisco.com/c/en/us/products/servers-unified-computing/ucs-m-series-modular-servers/index.html,.

[4] Coefficient of variation. https://en.wikipedia.org/wiki/Coefficient_of_variation.

[5] Dell and the Value of Density Whitepaper. http://i.dell.com/sites/doccontent/shared-content/data-sheets/en/Documents/ESG-Dell-and-The-Value-of-Density-Whitepaper.pdf.

[6] Dell-Blogs: The Time is Right - Introducing "Copper". http://en.community.dell.com/dell-blogs/direct2dell/b/direct2dell/archive/2012/05/29/arm-processor-based-dell-servers-on-the-way.

[7] Dell FX converged architecture components. http://www.dell.com/us/business/p/poweredge-fx/pd.

[8] Dell, HP, Cisco Look Beyond The Traditional Blade/Rack Server. http://www.crn.com/news/data-center/300078073/dell-hp-cisco-look-beyond-the-traditional-blade-rack-server.htm, Year = 2015,.

[9] High density and performance servers. http://www.mitac.com/business/Datun.html.

[10] HP MoonShot and AMD: Hosted Desktop in Healthcare: New Options. New Potential. https://www.amd.com/Documents/HDI-Healthcare-Whitepaper.pdf, Year = 2014,.

[11] HP Proliant m350 Server Cartridge. http://www8.hp.com/us/en/products/proliant-servers/product-detail.html?oid=7398903#!tab=features.

[12] HP Proliant m700 Server Cartridge. http://www8.hp.com/us/en/products/proliant-servers/product-detail.html?oid=6488207.

[13] HP Proliant m710p Server Cartridge. http://www8.hp.com/us/en/products/proliant-servers/product-detail.html?oid=8732043#!tab=specs.

[14] HP Proliant m800 Server Cartridge. http://www8.hp.com/us/en/products/proliant-servers/product-detail.html?oid=6532018#!tab=features.

[15] HPE Moonshot System. https://www.hpe.com/us/en/servers/moonshot.html.

[16] IDC: Hyperscale Data Centers Drive Server Growth in Q1 2015. http://www.eweek.com/servers/hyperscale-data-centers-drive-server-growth-in-q1-idc.html.

[17] IDC: Worldwide Server Market Revenues Increase 5.2% in the Fourth Quarter 2016 as Demand from China Once Again Drives Market Forward. http://www.idc.com/getdoc.jsp?containerId=prUS41076116.

[18] Intel Atom Processor N570. http://ark.intel.com/products/55637/Intel-Atom-Processor-N570-1M-Cache-1_66-GHz.

[19] Intel Xeon Processor D-1500 Product Family Product Brief. https://www.intel.com/content/www/us/en/processors/xeon/xeon-processor-d-brief.html.

[20] Intel Xeon Processor E3-1284L v4. http://ark.intel.com/products/88045/Intel-Xeon-Processor-E3-1284L-v4-6M-Cache-2_90-GHz.

[21] Intel Xeon Processor E3-1284L v4. http://ark.intel.com/products/77987/Intel-Atom-Processor-C2750-4M-Cache-2_40-GHz.

[22] List of Intel Xeon Microprocessors: Haswell-EP. https://en.wikipedia.org/wiki/List_of_Intel_Xeon_microprocessors.

[23] QCT Rackgo X Yosemite Valley. http://www.opencompute.org/products/rackgo-x-yosemite-valley/.

[24] SeaMicro SM15000 Fabric Compute Systems. http://www.seamicro.com/sites/default/files/SM15000_Datasheet.pdf.

[25] Sunon: How to select the right fan or blower. http://www.sunon.com/uFiles/file/03_products/07-Technology/004.pdf.

[26] System Overview for the SM15000 Family. http://www.seamicro.com/sites/default/files/SM_TO02_64_v2\%205.pdf.

[27] TI Keystone II Processor. http://www.ti.com/product/66AK2H12.

[28] Windows performance analysis. http://msdn.microsoft.com/en-us/performance/cc825801.

[29] The story behind the Active Cool fan design for the HP BladeSystem. In *HP ISS Technology Update Volume 7, Number 2*, 2008.

[30] ASHRAE Technical Committee 9.9 Mission Critical Facilities, Data Centers, Technology Spaces and Electronic Equipment: 2011 Thermal Guidelines for Data Processing Environments Expanded Data Center Classes and Usage Guidance., 2011.

[31] PCMark 7 PC performance testing white paper, 2011. http://www.futuremark.com/benchmarks/pcmark.

[32] Reflections on the Open Compute Summit., 2011. https://www.facebook.com/note.php?note_id=10150210054588920.

[33] BIOS and Kernel Developer's Guide (BKDG) for AMD Family 15h Models 00h-0Fh Processors, Jan 2012.

[34] HP Project Moonshot and the Redstone Development Server Platform, 2012. http://h10032.www1.hp.com/ctg/Manual/c03442116.pdf.

[35] A closer look at the Kabini die., 2013. http://www.anandtech.com/show/6977/a-closer-look-at-the-kabini-die.

[36] BIOS and Kernel Developer's Guide (BKDG) for AMD Family 16h Models 00h-0Fh Processors, May 2013.

[37] Cisco Global Cloud Index: Forecast and Methodology, 2014-2019 White Paper., 2016. http://www.cisco.com/c/en/us/solutions/collateral/service-provider/global-cloud-index-gci/Cloud_Index_White_Paper.html.

[38] SPECpower_ssj2008 Results., 2016. https://www.spec.org/power_ssj2008/results/.

[39] M. Arora, S. Manne, I. Paul, N. Jayasena, and D. M. Tullsen. Understanding idle behavior and power gating mechanisms in the context of modern benchmarks on CPU-GPU Integrated systems. *HPCA*, 2015.

[40] R. Ayoub, K. Indukuri, and T. Rosing. Temperature aware dynamic workload scheduling in multisocket CPU servers. *Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on*, 30(9):1359–1372, 2011.

[41] R. Ayoub, R. Nath, and T. Rosing. JETC: Joint energy thermal and cooling management for memory and CPU subsystems in servers. In *HPCA*, pages 1–12. IEEE, 2012.

[42] R. Ayoub and T. S. Rosing. Cool and save: cooling aware dynamic workload scheduling in multi-socket cpu systems. In *Design Automation Conference (ASP-DAC), 2010 15th Asia and South Pacific*, pages 891–896. IEEE, 2010.

[43] R. Z. Ayoub and T. S. Rosing. Predict and act: dynamic thermal management for multi-core processors. In *ISLPED*, pages 99–104. ACM, 2009.

[44] C. Bash and G. Forman. Cool job allocation: Measuring the power savings of placing jobs at cooling-efficient locations in the data center. In *USENIX Annual Technical Conference*, volume 138, page 140, 2007.

[45] L. Benini, A. Bogliolo, and G. De Micheli. A survey of design techniques for system-level dynamic power management. *IEEE transactions on very large scale integration (VLSI) systems*, 8(3):299–316, 2000.

[46] W. L. Bircher and L. John. Predictive power management for multi-core processors. In *ISCA*, 2012.

[47] S. Biswas, M. Tiwari, T. Sherwood, L. Theogarajan, and F. T. Chong. Fighting fire with fire: modeling the datacenter-scale effects of targeted superlattice thermal management. In *ACM SIGARCH Computer Architecture News*, volume 39, pages 331–340. ACM, 2011.

[48] M. T. Bohr, R. S. Chau, T. Ghani, and K. Mistry. The high-k solution. *IEEE spectrum*, 44(10):29–35, 2007.

[49] D. Brooks and M. Martonosi. Dynamic thermal management for high-performance microprocessors. In *HPCA*, pages 171–182. IEEE, 2001.

[50] J. Choi, Y. Kim, A. Sivasubramaniam, J. Srebric, Q. Wang, and J. Lee. Modeling and managing thermal profiles of rack-mounted servers with thermostat. In *HPCA*. IEEE, 2007.

[51] A. K. Coskun, D. Atienza, T. S. Rosing, T. Brunschwiler, and B. Michel. Energy-efficient variable-flow liquid cooling in 3D

stacked architectures. In *DATE*, 2010.

[52] A. K. Coskun, T. S. Rosing, and K. C. Gross. Proactive temperature management in MPSoCs. In *ISLPED*, 2008.

[53] A. K. Coskun, T. S. Rosing, and K. C. Gross. Utilizing predictors for efficient thermal management in multiprocessor SoCs. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 28(10):1503–1516, 2009.

[54] A. K. Coskun, T. v. Rosing, K. A. Whisnant, and K. C. Gross. Static and Dynamic Temperature-aware Scheduling for Multiprocessor SoCs. *IEEE Trans. Very Large Scale Integr. Syst.*, 16(9):1127–1140, Sept. 2008.

[55] A. K. Coskun, R. Strong, D. M. Tullsen, and T. Simunic Rosing. Evaluating the impact of job scheduling and power management on processor lifetime for chip multiprocessors. In *SIGMETRICS*, volume 37, 2009.

[56] J. Donald and M. Martonosi. Techniques for multicore thermal management: Classification and new exploration. In *ACM SIGARCH Computer Architecture News*, volume 34, pages 78–88. IEEE Computer Society, 2006.

[57] M. Gomaa, M. D. Powell, and T. N. Vijaykumar. Heat-and-run: Leveraging SMT and CMP to Manage Power Density Through the Operating System. In *ASPLOS*, pages 260–270, New York, NY, USA, 2004. ACM.

[58] S. Heo, K. Barr, and K. Asanović. Reducing power density through activity migration. In *ISLPED*, 2003.

[59] W. Huang, M. R. Stan, K. Sankaranarayanan, R. J. Ribando, and K. Skadron. Many-core design from a thermal perspective. In *Design Automation Conference, 2008. DAC 2008. 45th ACM/IEEE*, pages 746–749. IEEE, 2008.

[60] 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 *Micro*, 2006.

[61] S. Kaxiras, Z. Hu, and M. Martonosi. Cache decay: exploiting generational behavior to reduce cache leakage power. *ACM SIGARCH Computer Architecture News*, 29(2):240–251, 2001.

[62] J. Mars, L. Tang, R. Hundt, K. Skadron, and M. L. Soffa. Bubble-up: Increasing utilization in modern warehouse scale computers via sensible co-locations. In *Proceedings of the 44th Annual IEEE/ACM International Symposium on Microarchitecture*, MICRO-44, pages 248–259, New York, NY, USA, 2011. ACM.

[63] J. D. Moore, J. S. Chase, P. Ranganathan, and R. K. Sharma. Making scheduling "cool": Temperature-aware workload placement in data centers. In *Proceedings of the 2005 USENIX Annual Technical Conference, April 10-15, 2005, Anaheim, CA, USA*, pages 61–75, 2005.

[64] I. Paul, S. Manne, M. Arora, W. L. Bircher, and S. Yalamanchili. Cooperative boosting: needy versus greedy power management. In *The 40th Annual International Symposium on Computer Architecture, ISCA'13, Tel-Aviv, Israel, June 23-27, 2013*, pages 285–296, 2013.

[65] M. Pedram and S. Nazarian. Thermal modeling, analysis, and management in VLSI circuits: principles and methods. *Proceedings of the IEEE*, 2006.

[66] K. Puttaswamy and G. H. Loh. Thermal Herding: Microarchitecture Techniques for Controlling Hotspots in High-Performance 3D-Integrated Processors. In *Proceedings of the 2007 IEEE 13th International Symposium on High Performance Computer Architecture*, HPCA '07, pages 193–204, Washington, DC, USA, 2007. IEEE Computer Society.

[67] E. Rotem, A. Naveh, D. Rajwan, A. Ananthakrishnan, and E. Weissmann. Power-Management Architecture of the Intel Microarchitecture Code-Named Sandy Bridge. In *IEEE Micro*, 2012.

[68] H. Sanchez, B. Kuttanna, T. Olson, M. Alexander, G. Gerosa, R. Philip, and J. Alvarez. Thermal management system for high performance PowerPC/sup TM/microprocessors. In *Compcon'97. Proceedings, IEEE*, pages 325–330. IEEE, 1997.

[69] R. K. Sharma, C. E. Bash, C. D. Patel, R. J. Friedrich, and J. S. Chase. Balance of power: Dynamic thermal management for internet data centers. *IEEE Internet Computing*, 9(1):42–49, 2005.

[70] T. Simunic, L. Benini, A. Acquaviva, P. Glynn, and G. De Micheli. Dynamic voltage scaling and power management for portable systems. In *Proceedings of the 38th annual Design Automation Conference*, pages 524–529. ACM, 2001.

[71] M. Skach, M. Arora, C.-H. Hsu, Q. Li, D. Tullsen, L. Tang, and J. Mars. Thermal time shifting: Leveraging phase change materials to reduce cooling costs in warehouse-scale computers. In *ACM SIGARCH Computer Architecture News*, volume 43, pages 439–449. ACM, 2015.

[72] K. Skadron, M. R. Stan, K. Sankaranarayanan, W. Huang, S. Velusamy, and D. Tarjan. Temperature-aware microarchitecture: Modeling and implementation. *ACM Trans. Archit. Code Optim.*, 1(1):94–125, Mar. 2004.

[73] A. Snavely and D. M. Tullsen. Symbiotic jobscheduling for a simultaneous mutlithreading processor. *SIGPLAN Not.*, 35(11):234–244, Nov. 2000.

[74] J. Srinivasan and S. V. Adve. Predictive dynamic thermal management for multimedia applications. In *Proceedings of the 17th annual international conference on Supercomputing*, pages 109–120. ACM, 2003.

[75] M. R. Stan, K. Skadron, M. Barcella, W. Huang, K. Sankaranarayanan, and S. Velusamy. Hotspot: A dynamic compact thermal model at the processor-architecture level. *Microelectronics Journal*, 34(12):1153–1165, 2003.

[76] Q. Tang, S. K. Gupta, D. Stanzione, and P. Cayton. Thermal-aware task scheduling to minimize energy usage of blade server based datacenters. In *Dependable, Autonomic and Secure Computing, 2nd IEEE International Symposium on*, pages 195–202. IEEE, 2006.

[77] Q. Tang, S. K. Gupta, and G. Varsamopoulos. Thermal-aware task scheduling for data centers through minimizing heat recirculation. In *Cluster Computing, 2007 IEEE International Conference on*, pages 129–138. IEEE, 2007.

[78] Q. Tang, S. K. S. Gupta, and G. Varsamopoulos. Energy-efficient thermal-aware task scheduling for homogeneous high-performance computing data centers: A cyber-physical approach. *IEEE Transactions on Parallel and Distributed Systems*, 19(11):1458–1472, 2008.

[79] N. Tolia, Z. Wang, P. Ranganathan, C. Bash, M. Marwah, and X. Zhu. Unified thermal and power management in server enclosures. In *ASME 2009 InterPACK Conference collocated with the ASME 2009 Summer Heat Transfer Conference and the ASME 2009 3rd International Conference on Energy Sustainability*, pages 721–730. American Society of Mechanical Engineers, 2009.

[80] L. Wang, G. Von Laszewski, J. Dayal, X. He, A. J. Younge, and T. R. Furlani. Towards thermal aware workload scheduling in a data center. In *Pervasive Systems, Algorithms, and Networks (ISPAN), 2009 10th International Symposium on*, pages 116–122. IEEE, 2009.

[81] I. Yeo, C. C. Liu, and E. J. Kim. Predictive dynamic thermal management for multicore systems. In *Proceedings of the 45th annual Design Automation Conference*, pages 734–739. ACM, 2008.

[82] X. Zhou, Y. Xu, Y. Du, Y. Zhang, and J. Yang. Thermal management for 3D processors via task scheduling. In *Parallel Processing, 2008. ICPP'08. 37th International Conference on*, pages 115–122. IEEE, 2008.