The Impact of CPU Voltage Margins on Power-Constrained Execution

Panos Koutsovasilis, Christos D. Antonopoulos, Nikolaos Bellas, Spyros Lalis, George Papadimitriou, Athanasios Chatzidimitriou, Dimitris Gizopoulos,

Abstract—CPUs typically operate at a voltage which is higher than what is strictly required, using voltage margins to account for process variability and anticipate any combination of adverse operating conditions. However, these worst-case scenarios occur rarely, if ever, thus the operating voltage is overly pessimistic resulting in excessive power dissipation which leads to decreased performance under power capping. In this paper, we investigate the impact of reducing voltage margins beyond the nominal level on the efficiency of CPU power capping mechanisms, for three commercial systems, two Applied Micro ARMv8 micro-servers (X-Gene2 and X-Gene3) and an Intel x86-64 (Xeon E3). We show that CPU power capping at reduced voltage margins compared with Intel's RAPL and Dynamic Frequency Scaling (DFS) mechanisms results in performance improvement by up to 64% and 24% on average, respectively. In combination with state-of-the-art thread packing, the reduction of CPU voltage margins results in 36%, 33% and 27% performance improvement compared with RAPL and DFS for the Xeon E3 and the X-Gene processors, respectively. Also, we validate the robustness of our approach with a set of long-running experiments and show that significant energy gains can be achieved even when considering the cost of checkpointing and recovery in large-scale systems.

Index Terms—CPU power capping, CPU voltage margins reduction, Performance of Systems, Power Management

1 Introduction

Modern datacenters and high performance computing (HPC) systems are expected to operate under a tight power budget due to cost, power delivery, and cooling limitations. This is a challenging undertaking, as in the past decade power and cooling costs have doubled [1]. In particular, CPUs account for up to 60% of the total power consumption of compute nodes [2], whereas datacenter cooling has a power footprint equal to that of the compute infrastructure. An important exercise for both the datacenter and HPC domains is to identify and apply techniques that optimize throughput/performance in a power-constrained environment.

Power capping is a technique commonly used by datacenter and HPC systems administrators to meet a power threshold by reducing the operating frequency of the CPU (CPU throttling) [3], [4], [5], [6], [7], [8]. This may allow for a larger number of server nodes to be deployed (increasing datacenter capacity) without exceeding the aggregate power budget [9]. However, the performance penalty due to frequency downscaling may violate the Quality-of-Service (QoS) agreement between the operator and the customer in a cloud datacenter [10], or may even be prohibitive in an HPC environment.

State-of-the-art research efforts [6], [7], [8] in CPU power capping propose hybrid techniques that utilize both hardware and software mechanisms to meet a certain power cap while minimizing the performance penalty. All these works support only Intel CPUs and utilize Running Average Power Limit (RAPL) [11], which is a hardware mechanism that limits CPU power consumption under a cap. However, RAPL itself misses some opportunities for further performance improvement, as the operating points of the CPU are chosen conservatively. In particular, to account for manufacturing variability, adverse external conditions, and varying workload pressure, CPUs operate at a voltage which is higher than what is strictly required for any given operating frequency to guarantee fault-free operation. Yet, prior research [12], [13], [14], [15], [16], [17], [18], [19], [20] suggests that these voltage margins (guardbands) are overly pessimistic.

In this paper, we study the effects of carefully reducing those CPU voltage guardbands in power capped environments. We focus on CPU power consumption, power consumption at the node-level (at the plug), performance/throughput and CPU temperature. We compare power capping when operating at reduced margins against conventional hardware, software, and hybrid (a combination of hardware & software) approaches under nominal voltage margins. We also investigate whether operation at reduced margins can be combined with conventional power capping approaches.

Our evaluation involves three different off-the-shelf CPUs, namely an Intel x86-64 (Xeon E3-1220v5) and two ARMv8 (AppliedMicro X-Gene 2 and X-Gene 3), which are used to run a mix of benchmarks from various suites, such as CloudSuite [21], Parsec [22], SPEC CPU2006 [23], as well as various stress-tests [24], [25], [26].

Our experimental study shows that reducing CPU voltage margins can significantly improve performance, reduce CPU & node power consumption, as well as CPU temperature. More specifically, reduction of voltage margins re-

- *P. Koutsovasilis, C. D. Antonopoulos, N. Bellas and S. Lalis are with the Department of Electrical and Computer Engineering, University of Thessaly, Greece.E-mail: {pkoutsovasilis, cda, nbellas, lalis}@uth.gr*
- *G. Papadimitriou, A. Chatzidimitriou and D. Gizopoulos are with the Department of Informatics and Telecommunications, University of Athens Greece.E-mail: {georgepap, achatz, dgizop}@di.uoa.gr*

Manuscript received December 05, 2019; revised May 16, 2020.

sults in improved performance compared with conventional power capping approaches by 64%, 30% and 34% on average for Xeon E3, X-Gene 2 and X-Gene 3, respectively. Also, it outperforms hybrid power capping solutions, namely PUPiL [7], by 24%, 22% and 5% on average for Xeon E3, X-Gene 2 and X-Gene 3, respectively.

Furthermore, to validate in practice the reliability of nodes operating at reduced voltage margins, we execute random, mixed workloads on 16 Xeon E3-based nodes that operate at reduced voltage margins, for 23 days without observing any errors. Based on these results, we show that the energy gains by operation at reduced voltage margins remain significant, even in large-scale deployments where the increased probability of failure may necessitate stronger protection, such as more frequent checkpointing.

Although CPU power capping has been researched quite extensively over the last years [3], [5], [6], [7], [8], [27], [28], as shown in Table 1, no previous work investigates the effects of reducing conservative CPU voltage guardbands to optimize execution in a power-constrained environment.

This paper makes the following contributions:

- To the best of our knowledge, this is the first work that studies the interplay between CPU power capping and operation at reduced margins with respect to various metrics and across multiple platforms.
- We experimentally, on real systems, demonstrate that conventional CPU power capping mechanisms can be combined with operation at reduced voltage margins.
- We introduce Reduced Voltage Scaling power Capping (RVSCap), a novel CPU power capping mechanism that operates the CPU at reduced voltage margins, minimizes the performance penalty of power capping at restrictive caps, reduces the power footprint at more generous caps and supports multiple platforms.
- This is the first work that experimentally evaluates the effects of CPU operation at reduced voltage margins with long experiments and statistically estimates the respective effects on large-scale deployments.

The rest of the paper is organized as follows. In Section 2 we present an example motivating CPU operation at reduced voltage margins for power-constrained execution.

TABLE 1: Overview of this work and comparison to prior approaches.

Approaches /Features	This work	[3], [5], [6], [7], [8], [27], [28]	[12], [13], [14], [15], [16], [17], [18], [19], [20]
CPU power capping	✓	✓	✗
CPU voltage margins reduction	✓	✗	✓
Multi-platform	✓	✗	✗
Effects on large-scale deployments	✓	-	✗

Section 3 provides an overview of the CPUs used in our study. Section 4 introduces the power capping mechanisms (conventional and new) that we evaluate. In Section 5, we present the results of our experimental study, while Section 6 presents the expected energy gains on large-scale deployments. Section 7 provides an overview of related work. Finally, Section 8 concludes the paper.

2 Motivation

Figure 1 depicts the power consumption of *bzip2* from the SPEC CPU2006 benchmark suite [23] for three different execution scenarios: one without a CPU power cap (*dark red* line) and two at a CPU power cap of 20W at nominal (*orange* line) and at reduced CPU voltage margins (*green* line). The experiments have been performed on a Xeon E3-based node (see Table 2) which has a Thermal Design Power (TDP) of 80W, and, thus, the 20W power cap is rather restrictive. We concurrently run multiple instances of *bzip2* to fully engage all four cores of the CPU. The horizontal distance between the ending points of the *dark red* and *orange* vertical lines indicates the performance penalty due to CPU power capping. Similarly, the horizontal distance between the ending points of the *orange* and *green* vertical lines represents the performance gains of CPU voltage margins reduction, compared with nominal CPU operation for the same cap.

In the power-constrained execution with nominal CPU operation, the clock frequency is reduced significantly to meet the cap, leading to a 36% drop in performance. On the other hand, with CPU operation at reduced voltage margins, the 20W power cap is met without underscaling the frequency as much, and hence, with substantially lower performance degradation. More specifically, for the power-constrained execution at nominal voltage, the CPU frequency is 1.9 GHz on average and for the execution at reduced voltage margins (−170 mV reduction) the average CPU frequency is 2.5 GHz. As a result, we observe 22% improved performance at reduced voltage margins compared with capped execution at nominal voltage. These results demonstrate the opportunity of harnessing voltage margins to meet restrictive power caps without resorting to extreme frequency downscaling.

3 Platform Overview

3.1 Architectural Characteristics

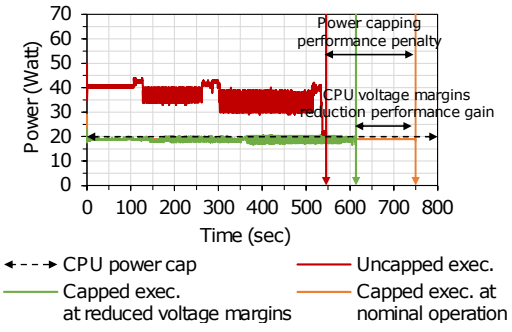

Fig. 1: Impact of reducing CPU voltage margins on performance when the CPU operates at a power cap.

In our study, we use three multicore CPUs: two different ARMv8 CPUs and an Intel CPU. Table 2 summarizes the main characteristics of these CPUs.

The two ARMv8-based platforms are the 8-core X-Gene 2 and 32-core X-Gene 3 from AppliedMicro (APM). These CPUs include a dedicated Power Management processor (PMpro) [29], which is controlled by the Scalable Lightweight Intelligent Management processor (SLIMpro) [29]. PMpro exposes advanced power management capabilities and can be used to configure CPU attributes such as the operating voltage. All CPU cores and caches have a common voltage domain, with the exception of the L3 cache in X-Gene 2, which has a separate voltage domain that, however, cannot be down-scaled. These capabilities are monitored and managed by the SLIMpro, which is accessed from the Linux kernel through an I^2C interface. Both processors support frequency scaling through the Collaborative Processor Performance Control (CPPC) power and performance management interface [30]. The frequency on both X-Gene processors can be adjusted per pair of cores.

The Intel CPU we use in our study is the Xeon E3-1220v5. Like other modern Intel CPUs, it implements two power management mechanisms in hardware, namely the Fully Integrated Voltage Regulator (FIVR) [31] and Speedshift [32]. FIVR exposes certain Model-Specific Registers (MSRs) through which the OS can request to apply an offset over the nominal operating voltage of the CPU. All CPU cores and caches share a common voltage domain. Speedshift enables fully autonomous control of P-State selection by the CPU, known as Hardware P-States (HWP), supporting a dynamic, wide power range with faster transition times between the minimum and maximum supported frequencies. All CPU cores are clocked with the same frequency. The OS may disable autonomous scaling by setting the maximum and minimum frequencies of a P-State to the same value. Also, Intel CPUs feature the RAPL mechanism which can limit the power consumption of each component within a cap. Furthermore, RAPL provides information about the power consumption of different components of the system such as the CPU, memory, etc.

3.2 Characterization of Voltage Margins

In this study, we investigate the interplay between reducing CPU voltage margins and power capping mechanisms. To this end, even though this is not the focus of our work, it is necessary to quantify the voltage margins of the target platforms through a characterization process. We apply the methodology proposed in prior work [12], [13], [17], [18], [19], for the CPUs of Table 2, using SPEC CPU 2006 [23], Parsec [22] and micro-virus [19] benchmarks.

TABLE 2: Characteristics of the three target CPUs.

Characteristic	Xeon E3-1220v5	X-Gene 2	X-Gene 3
CPU Cores	4	8	32
Base clock freq.	3.0GHz	2.4GHz	3.0GHz
Lowest clock freq.	0.8GHz	0.3GHz	0.375GHz
Manufacturer	Intel	APM	APM
Family	Skylake	ARMv8	ARMv8
Technology	14nm	28nm	16nm
Max Performance	4 (issue-slots/cycle)	4 (IPC)	4 (IPC)
Freq. Control	HWP	CPPC	CPPC
TDP (W)	80	35	125

More specifically, for each CPU frequency point we run each benchmark 20 times at each voltage step, in the entire range from the nominal voltage point down to the lowest sub-nominal voltage, referred as V_{min}, at which *any* workload executes successfully without any hardware reported error, silent data corruption (SDC), or system crash. As an additional robustness test for the value of V_{min}, we execute each benchmark 1000 more times and verify that all runs finish successfully. For SDC detection we compare the output of each workload under reduced voltage margins with the corresponding one by nominal execution. This V_{min} quantification methodology is conservative because it misses workload-dependent opportunities to further reduce V_{min}, yet we opt to potentially sacrifice some additional power efficiency, in order to mitigate and preclude any erratic behavior, such as SDCs, when voltage margins are aggressively reduced [13], [17], [19].

Figure 2 illustrates the V_{min} identified by our characterization for a range of operating frequencies for all three CPUs. The grey area shows a configuration where — during our extensive characterization — at least one workload did not execute successfully. The point at the boundary of the green area (safe operation) and grey area (unsafe operation) corresponds to V_{min}. For the Xeon E3 processor, which supports DVFS, we observe that V_{min} drops with frequency reduction. In terms of absolute values, the width of voltage margins ranges from 240mV to 170mV depending on the CPU operating frequency. The observation that the width of voltage margins varies is critical, as the voltage manipulation mechanism of the Xeon E3 processor is offset-based; the different width of voltage margins across different CPU frequencies requires extra care to be taken when transitioning between frequencies.

For the X-Gene 2 and X-Gene 3 processors, which only support dynamic frequency scaling (DFS), the nominal operating voltage is constant for all frequencies, at 980mV and 875mV, respectively. Moreover, major differences in V_{min}

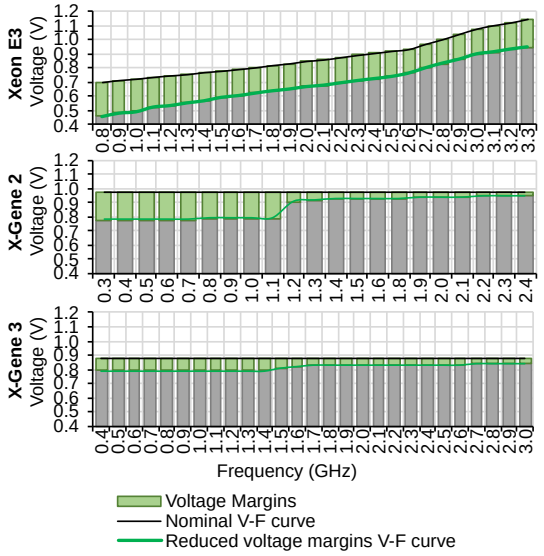

Fig. 2: Voltage (y-axis) vs. frequency (x-axis) for each CPU.

can be observed only after reducing the frequency to half of that used by the highest performance operating point. Both X-Gene processors use clock skipping in combination with clock division to generate the effective frequency of the processor cores. Whenever the target frequency is less than half of the maximum clock frequency, clock division is first applied. All other intermediate frequency points are implemented with clock skipping, either on the original or on the divided clock frequency. This explains why we do not observe any variation of V_{min} at intermediate frequencies, with relatively large V_{min} differences at frequencies where clock division is applied. Voltage margins are, again, wider at lower frequencies.

The outcome of this characterization (shown in Figure 2) is an extended voltage-frequency curve for each CPU, which, for each frequency, provides the sub-nominal voltage V_{min} deemed to be safe by the characterization campaign for all workloads.

We note that similar to the aforementioned characterization work, aging effects of the CPUs are out of the scope of this study. Also, we do not investigate system operation at extreme temperatures. Prior research [33], [34] indicates that CPU voltage margins are wider at higher temperatures, due to the transistor-level effect of temperature inversion. However, other components such as disks [35] and DRAMs [36] are more failure-prone at higher temperatures, which makes it difficult to reason about node reliability at extreme temperature conditions. For this reason, in our experiments we operate all our machines under typical data center conditions [37] (at an ambient temperature of 21-23°C), despite the fact that this is expected to result in narrower CPU voltage margins.

4 Power Capping Approaches

4.1 Existing Techniques

There are numerous software-based capping approaches in the literature [3], [5], [27], [28], which use CPU performance counters to monitor key performance metrics. These metrics are then used to calculate the CPU frequency downscaling required to enforce the cap.

Modern Intel CPUs, including the Xeon E3 used in our study, feature RAPL [11], a hardware power capping mechanism that directly observes the power consumption of the CPU. The time window of cap enforcement is user-specified. RAPL samples CPU power consumption within the window and solves a linear equation [7], [11] in order to dynamically decide and apply the most suitable frequency and voltage pair from the DVFS operating points supported by the CPU. For the Xeon E3, the cap enforcement time window can be as narrow as 1 msec. On the other hand, the X-Gene processors do not provide a hardware capping mechanism.

PUPiL [7] is a recent technique for maximizing the performance of multithreaded applications on Intel x86-64 systems. PUPiL uses RAPL for power capping and manipulates thread allocation to cores based on a heuristic thread-packing algorithm for identifying the best performing threads-to-cores ratio, where the number of threads is larger than the number of cores. PUPiL measures performance indirectly using heartbeats [38] which are generated by the executing workloads.

4.2 RAPL-RM

In order to investigate the impact of exploiting voltage margins on power capped environments, we investigate the combination of the existing RAPL mechanism with reduced voltage margins (RAPL-RM). Similar to RAPL, RAPL-RM supports only Intel CPUs. Given that RAPL switches between frequencies under hardware control, without any notification to the software, it is not possible in RAPL-RM to adjust the voltage offset according to the width of the voltage margin at each operating frequency. Instead, we conservatively apply the safest Voltage offset that corresponds to the narrowest voltage margin (170mV) across all frequencies, missing the opportunity to further decrease the voltage by up to 70mV at specific frequencies.

4.3 RVSCap

To maximize the power gains introduced by reducing CPU voltage margins and hence minimize the performance penalty induced by a power cap, we design and implement Reduced Voltage Scaling power Capping (RVSCap), a software-based power capping mechanism that reduces CPU voltage by shaving-off pessimistic margins. Also, unlike RAPL and the RAPL-RM extension discussed above, RVSCap is a software-based mechanism that can be employed in widely different platforms, merely by implementing a few platform-specific components.

Similarly to RAPL and RAPL-RM, RVSCap is a closed-loop mechanism that measures the average power consumption of the CPU at the boundaries of time windows, identifies the CPU operating point that satisfies the specified power cap (based on the equation $P = CV^2F$), and then enforces that operating point for the following time window. However, in contrast to RAPL and RAPL-RM, RVSCap includes all the necessary logic mentioned in Section 3.2 for safe transition between different $V_{min} - F$ pairs. This is critical as the width of the voltage margin varies across CPU frequencies and a premature adjustment of CPU frequency without properly setting the V_{min} will result in a system crash.

As shown in Figure 3, when RVSCap needs to take action because the current power consumption (as reported by the CPU at execution time, e.g. through RAPL on Intel and SLIMPro on X-Genes) is above the defined CPU power threshold, it first checks if the cap can be met by scaling the CPU voltage to a sub-nominal, yet safe level for the current operating frequency, based on the V_{min} quantified via the offline characterization (Section 3.2). If the estimated power reduction by voltage scaling is not sufficient, RVSCap re-calculates the power consumption for the immediately lower operating frequency and the corresponding V_{min}. This is repeated, in an iterative fashion, until an operating point is found that meets the desired power cap, which is then applied to the CPU.

In the opposite case, when the current power consumption is below the threshold, RVSCap calculates and applies the operating point with the highest frequency that meets the power cap. In case RVSCap applies the maximum CPU frequency, the exploitation of voltage margins translates to lower CPU power consumption. This additional power budget slack can be used to power-on additional nodes,

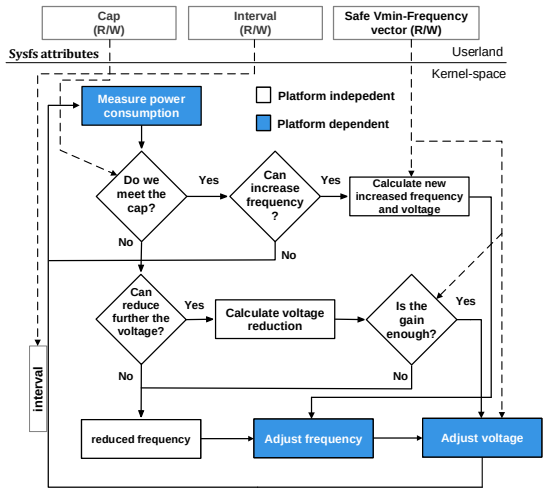

Fig. 3: Flowchart of RVSCap. Only a few components (in blue) need to be custom-developed for the target platform.

thus increasing the active computational capacity of the datacenter or can be exploited towards reducing power and cooling costs.

RVSCap is a simple, lightweight mechanism, that supports multiple platforms and can be used by any user-level runtime system. More specifically, RVSCap can support any platform that exposes the CPU power consumption and allows to adjust independently the operating frequency and voltage of the CPU. To port RVSCap to a new platform, one only needs to implement a small number of platform-dependent components that provide the high-level (platform-independent) API used by the rest of the mechanism (see Figure 3).

The effective cap enforcement interval depends on the low-level mechanisms that are available in the underlying platform. We experimentally find that for the Intel Xeon E3 CPU an effective interval of cap enforcement can be as low as 10 msec with 0.29% overhead. On the other hand, this interval is much higher, at approximately 1 sec, for the X-Gene platforms. This is due to the fact that every request of the capping mechanism (reading the CPU power consumption or setting the CPU operating frequency) has to go through an I^2C interface to the SLIMpro processor and then to the PMpro processor.

5 Experimental Study

In this section, we discuss the results of a detailed experimental study of the effects of exploiting reduced voltage margins during power-constrained execution. Moreover, we combine execution at reduced voltage margins with other state-of-the-art power-capping approaches, such as RAPL and PUPiL and compare against the execution at nominal operating points.

The study is conducted using three different platforms that feature the CPUs in Table 2. We discuss results in terms of performance, CPU and node power consumption, CPU operating temperature, as well as timeliness. We capture the CPU power consumption as reported through RAPL for Xeon E3 and through SLIMpro for X-Genes. The CPU operating temperature is reported through MSRs for Xeon E3 and through SLIMpro for X-Genes. Node power consumption is measured using an external power meter at the plug of each system.

Although the effectiveness of our approach depends on the width of the CPU voltage margins, which varies for different CPU models, we show that even for the worst case of processors that exhibit extremely narrow voltage margins, such as the X-Gene processors, the gains of combining operation at reduced voltage margins with power cap limits remain significant.

5.1 Benchmarks

We use 16 benchmarks from CloudSuite [21], Parsec [22], SPEC CPU2006 [23], and individual stress-tests [24], [25], [26] for FP computations, thermal load, and memory use, as shown in Table 3. Note that the CloudSuite benchmarks and the stress-tests are not a part of the characterization process, presented in Section 3.2. We include them in our experimental study, as an extra validation of the safety of V_{min} values identified during the characterization process. For parallel and sequential benchmarks we manipulate the degree of parallelism and the number of co-executing instances respectively, to achieve maximum core utilization and maximize the CPU power footprint with the exception

TABLE 3: Benchmark set.

Suite	Benchmark	Reference name	Baseline power consumption (W)			Throughput metric	Input
			Xeon E3	X-Gene 2	X-Gene 3		
Cloudsuite [21]	Data Analytics	danal	26	12	28	MB/sec	default
	Data Caching	dcach	42	20	33	requests/sec	default
	Web Search	wsrch	13	10	22	searchDriver	default
	Web Serving	wsrv	14	11	24	elggDrives	default
Parsec [22]	Blackscholes	blcks	34	23	70	prices/sec	native
	Facesim	fcsim	39	24	47	frames/sec	native
	Fluidanimate	fanim	41	25	65	steps/sec	native
	Swaptions	swpts	40	27	80	swapt./sec	native
SPEC CPU 2006 [23]	Bzip2	bzip2	39	26	83	MB/sec	ref
	Bwaves	bwves	32	26	83	steps/sec	ref
	Gromacs	grmcs	36	23	70	frames/sec	ref
	h264Ref	h264r	39	27	89	frames/sec	ref
Stress-test	Memstress	mste	31	21	65	Bytes/sec	default
	Linpack [24]	lpack	57	26	80	equat./sec	default
	Prime95 [25]	mprime	62	-	-	primes/sec	default
	Firestarter [26]	fires	63	-	-	fma op/sec	default

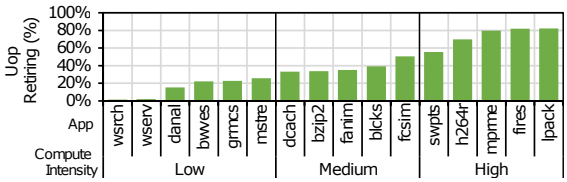

Fig. 4: Uops retiring(%) normalized wrt. CPU max performance (Table 2) on Xeon E3.

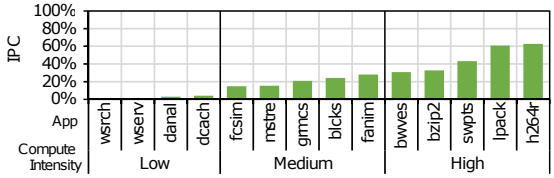

Fig. 5: Retired instructions per cycle (IPC)(%) normalized wrt. CPU max performance (Table 2) on X-Gene 3.

of CloudSuite benchmarks, which can not be scaled in a practical way.

To better understand the behavior of the benchmarks of Table 3 in a power-constrained execution environment, we categorize them into three groups according to their instruction retirement rate, thus, implicitly according to their compute intensity. Figure 4 shows the percentage of retired micro-operations (uops), averaged across all 4 CPU cores of the Xeon E3, for each benchmark. Following Intel's Top-Down Microarchitecture Analysis Method (TMAM) [39], the remaining uops are attributed to bad-speculation, as well as to back-end and front-end bound operations, which cover several types of stalls that prevent the delivered uops from retiring. Similarly, Figure 5 depicts the retired instructions per cycle (IPC) averaged across all 32 CPU cores of the X-Gene 3, for each benchmark. We omit the corresponding results for X-Gene 2 as they are similar to those of X-Gene 3.

Most benchmarks in CloudSuite have low compute intensity, as they do not utilize all the available CPU cores. Interestingly, there are benchmarks that exhibit different behavior on the Xeon E3 and X-Gene 3. For example, *facesim* is more compute-intensive than *Blackscholes* on the Xeon E3, whereas it turns out to be (far) less compute-intensive on the X-Gene 3, where its execution leads to a larger number of slow memory accesses. This is due to the higher thread contention for capacity at the last level cache of X-Gene In Xeon E3, 4 threads share 8MB of L3, whereas in X-Gene 3 all 32 threads share 32 MB of L3.

5.2 Effects of CPU Voltage Margins on Power Capping

In this section, we quantify the impact of exploiting reduced CPU voltage margins in power capped environments, for various CPU power capping mechanisms. On-chip power capping mechanisms offered by CPU manufacturers support capping at a wide range of power budgets. For example, RAPL on Intel CPUs, supports capping at any power limit within the TDP range of the chip. We opted to perform our study for an equally wide range of power caps across the three different CPUs we use in our study.

For the Xeon E3 CPU, we evaluate the following power capping techniques: i) RVSCap, ii) conventional RAPL, and iii) *RAPL-RM*. Figure 6 shows the results. The x-axis of each row represents the benchmarks under different CPU power caps, sorted according to their compute intensity (section 5.1). The left y-axis (and the corresponding bars) quantifies the absolute values of each metric. The two lines refer to the right y-axis and show the average per cap relative increase or reduction among all benchmarks of the metric attained by RAPL-RM and RVSCap over RAPL.

We observe that RVSCap outperforms both RAPL and RAPL-RM, and depending on the enforced cap the gains translate to improved performance or lower power consumption both at CPU and node level, as well as lower

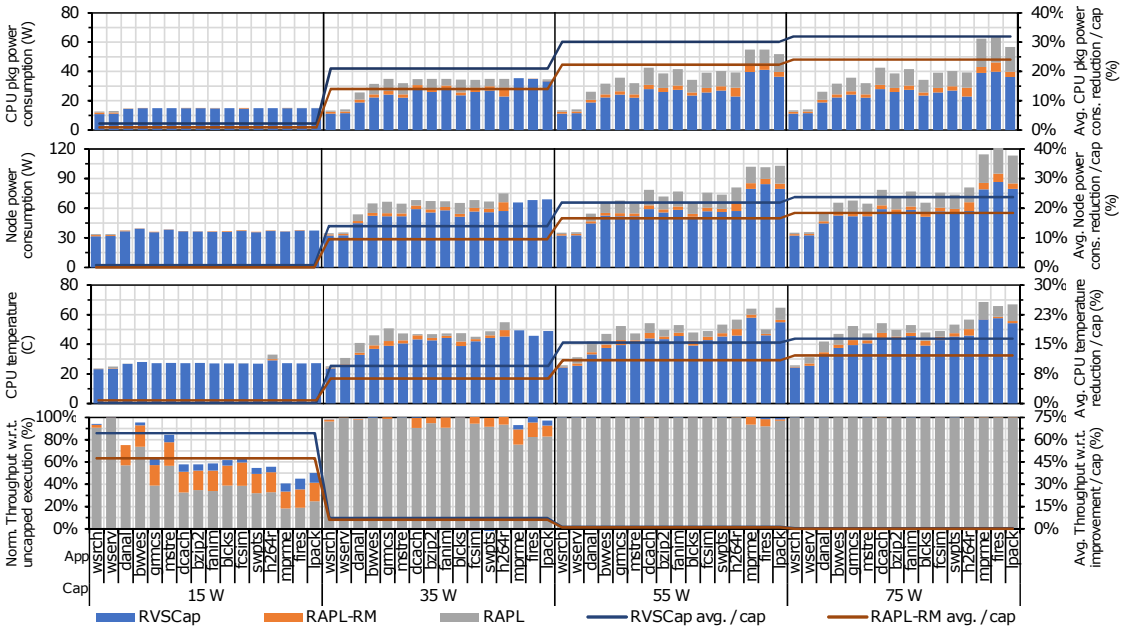

Fig. 6: CPU and total node power dissipation, CPU temperature and performance for different power caps and power capping mechanisms on Xeon E3. Bar plots correspond to the left y-axis and lines to the right y-axis.

CPU operating temperature. In general, the trend is that for aggressive power caps (15W - 18% of Xeon E3 TDP) the reduced voltage margins enable significantly improved performance. For higher caps (35W,55W - 43%,68% of Xeon E3 TDP), which are restrictive for fewer benchmarks, the voltage margins reduction results in both performance gains and reduction in power consumption and operating temperature of the CPU. Eventually, for higher caps (75W - 94% of Xeon E3 TDP) where conventional power capping typically does not result in performance penalties, voltage margins reduction results in lower power dissipation and CPU temperatures. As an example, RVSCap results in 104% higher performance for *Linpack* under aggressive power caps, while for relaxed power caps it reduces node power consumption and CPU operating temperature by 31% and 19% respectively, compared with RAPL. On average, depending on the enforced cap, RVSCap delivers up to 64% improved performance, 32% lower CPU power consumption, 24% lower power consumption at the node level and 16% lower CPU temperature, compared with RAPL.

Power consumption gains can only be observed when the workload is not constrained by the power cap (because its baseline – uncapped – power consumption is lower than the cap) and the CPU operates at its highest frequency. In this case, power gains are fully attributable to voltage margins reduction at the highest frequency operating point. In all other cases where the workload is constrained by the power-cap, the reduction of voltage margins (naturally) translates to performance improvement. Observing power gains in those cases would signify a sub-optimally operating mechanism, that fails to effectively utilize the power budget made available to it and, as a result, unnecessarily penalizes workload performance. Whether a given power cap is restrictive (thus the gains will be translated to performance), or not (thus the gains will be translated to power) depends on the workload. However, our experimental results indicate that the impact of the reduction of voltage margins on CPU power footprint effectively shifts towards lower caps the point where, for a given workload, gains begin to manifest as power savings rather than performance improvement. Assuming a stable cooling capacity, CPU temperature is directly connected to power dissipation, therefore the same observations hold for CPU temperature as well.

In comparison with RAPL-RM, RVSCap delivers on average up to 17% improved performance, 8% lower CPU power consumption, 6% lower node power consumption and 4% lower CPU operating temperature. These gains are due to the fact that RVSCap is able to dynamically apply the appropriate V_{min} for each CPU operating frequency, rather than using the safest V_{min} across all frequencies. Moreover, RVSCap can achieve a 55W power cap without performance throttling on *any* of the benchmarks. On the other hand, to respect the same cap RAPL and RAPL-RM have to throttle the performance in 3 and 2 of the 16 benchmarks, respectively.

Given that X-Gene processors do not support DVFS and do not offer any power capping mechanism, in order to present a comprehensive comparison with execution at reduced voltage margins enabled by RVSCap, we implement a software power capping mechanism that employs only frequency scaling to emulate a conventional frequency scaling power capping mechanism (DFSCap). More specifically, DFSCap has the same logic as RVSCap, however, it does not consider operating at reduced voltage margins points.

Figures 7 and 8 illustrate the performance of those mechanisms on the X-Gene 2 and X-Gene 3 CPUs, respectively. The trends are generally similar to what was observed on the Intel-based system. The requested power cap acts as a knob that directs the benefits introduced by the reduction of voltage margins towards either performance gains for aggressive caps, or reduced power consumption for higher, less restrictive power caps. Notably, the voltage margins

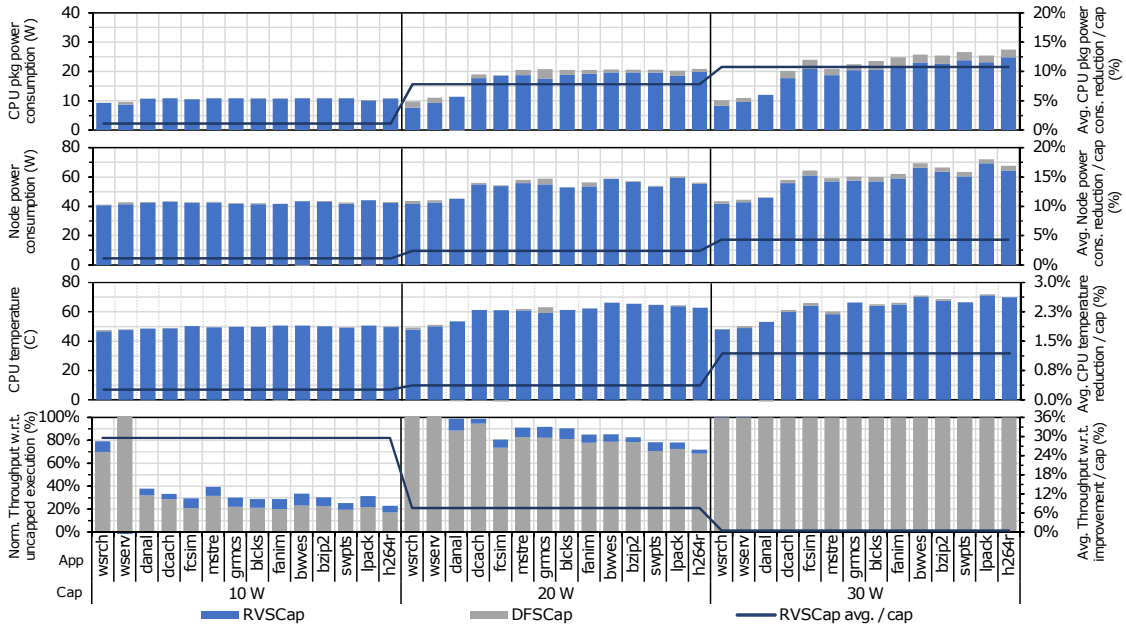

Fig. 7: CPU and total node power dissipation, CPU temperature and performance for different power caps and power capping mechanisms on X-Gene 2. Bar plots correspond to the left y-axis and lines to the right y-axis.

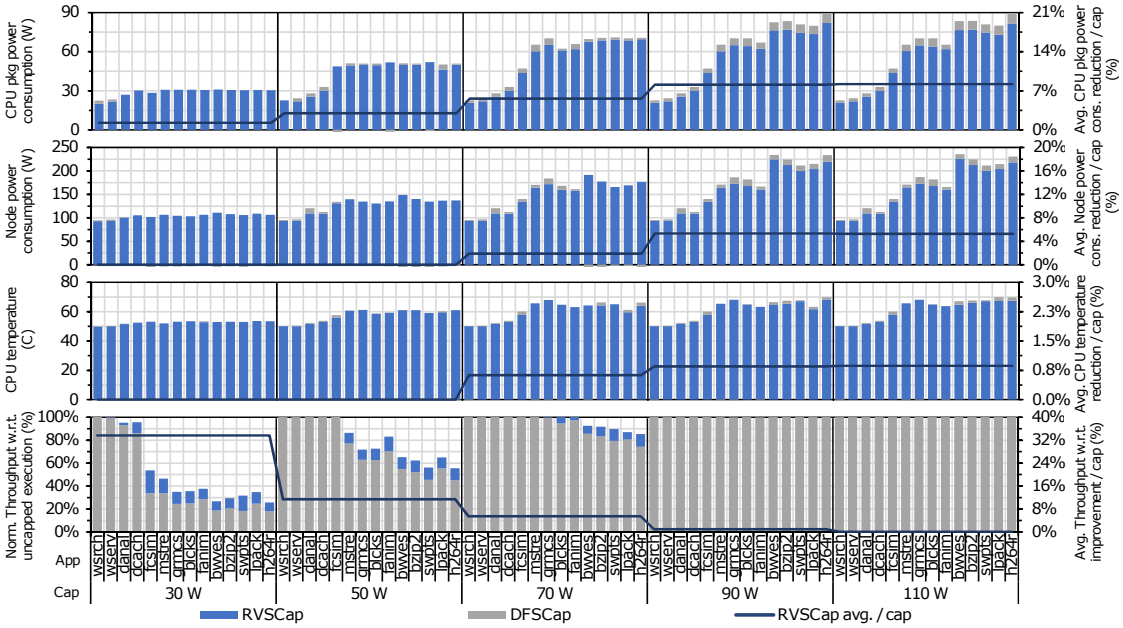

Fig. 8: CPU and total node power dissipation, CPU temperature and performance for different power caps and power capping mechanisms on X-Gene 3. Bar plots correspond to the left y-axis and lines to the right y-axis.

TABLE 4: Average gains of RVSCap under aggressive and relaxed power caps, for the different compute-intensity groups of the benchmarks.

Power Cap	Metric	Xeon E3			X-Gene 2			X-Gene 3		
		Low	*Medium*	*High*	*Low*	*Medium*	*High*	*Low*	*Medium*	*High*
aggressive	Performance improvement	30%	69%	102%	11%	36%	37%	3%	43%	49%
relaxed	CPU power reduction	26%	33%	37%	11%	12%	10%	9%	7%	8%
relaxed	Node power reduction	17%	25%	31%	3%	5%	5%	5%	5%	6%
relaxed	CPU temperature reduction	17%	16%	17%	1%	2%	2%	1%	2%	2%

reduction on X-Gene processors does not have a significant impact on CPU operating temperature, as was the case on Xeon E3. This is because the CPU fans on both X-Gene systems operate in two fixed distinct modes (low and high rate), depending on CPU temperature. On the contrary, for the Intel platform, the rate of the CPU fans scales accordingly with the CPU temperature.

On average, depending on the requested cap, RVSCap delivers up to 30% improved performance, 11% reduced CPU power consumption, 4% reduced node power consumption and 1% reduced CPU temperature for X-Gene 2 compared with DFSCap. On X-Gene 3, the respective gains are 34% improved performance, 8% lower CPU power consumption, 5% lower node power consumption and 1% reduced CPU temperature.

So far, our investigation clearly indicates that the exploitation of CPU voltage margins in a power capped environment (RVSCap), outperforms CPU capping mechanisms which control only CPU frequency. Table 4 summarizes the gains attained by RVSCap over RAPL for Xeon E3 and over DFSCap for X-Gene processors, averaged over the three compute intensity groups of benchmarks, under aggressive and relaxed power caps. We observe that on Xeon E3 the high-intensity benchmark group demonstrates the highest performance benefits for aggressive caps and the lowest power consumption and CPU operating temperature for relaxed caps. For the X-Gene processors we observe that, under aggressive power caps, the highest compute intensity group again has the highest performance gains.

Modern Intel processors, like the Xeon E3 used in our study, exploit *C-States*, a hardware mechanism which dynamically powers on or off different modules of the CPU, according to the resource requirements of the workload. In Table 4 we observe that thanks to this mechanism, on the Xeon E3, the reduction of voltage margins results – for relaxed power caps – in different energy gains for different intensity groups. However, the X-Gene processors do not feature a similar mechanism and, consequently, the reduction of voltage margins results in similar energy gains, irrespective of the compute-intensity of the benchmark.

Another important performance metric of power capping mechanisms is timeliness. Timeliness is the settling time required for CPU power to stabilize at the cap, after a new cap is requested. Figure 9 depicts the average timeli-

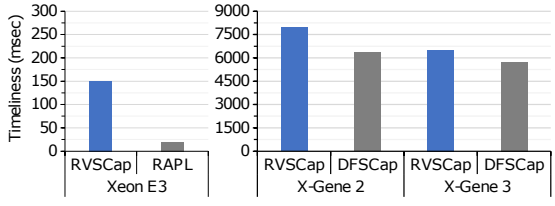

Fig. 9: Timeliness of RVSCap, RAPL, and DFSCap mechanisms for Xeon E3, and X-Gene processors.

ness, across all applications, for RVSCap (on all platforms), RAPL (on Intel) and DFSCap (on the two X-Gene platforms), for the 35W, 20W and 50W power caps on Xeon-E3, X-Gene 2 and X-Gene 3 processors, restrictively. For other caps the timeliness is similar.

On Xeon E3, the timeliness of RVSCap and RAPL is at 150msec and 20msec, respectively. This difference between RAPL and RVSCap is expected, as the latter is a software mechanism. That said, RVSCap significantly outperforms other software mechanisms in the literature [7] (Soft-Decision in the order of 95sec and soft-DVFS in the order of 7.3sec), and is also better than PUPiL (with a timeliness of 356msec). The latter, however, is a hybrid mechanism that directly uses RAPL for power control. Although the latter is a hybrid mechanism that directly uses RAPL for power control, its timeliness is not representative of the timeliness of RAPL. More specifically, the authors of [7] opt to setup RAPL to enforce the power limit at a coarse time granularity, in order to allow PUPiL sufficient time to apply thread-packing decisions before RAPL throttles the CPU.

For the X-Gene processors, the timeliness of both RVSCap and DFSCap is at the granularity of seconds. More specifically, the settling times for RVSCap and DFSCap on the X-Gene2 are 7.9sec and 6.4sec respectively. On the X-Gene 3 the respective values are somewhat lower, at 6.5sec and 5.7sec. This is because of the inherent delay, discussed in Section 4.3, for adjusting the operating points and for monitoring power consumption on the X-Gene processors. RVSCap exhibits slightly worse timeliness compared with DFSCap, as RVSCap controls both the CPU operating frequency and voltage, which translates to more instructions issued on the I^2C processor control interface. Still, the timeliness on the X-Gene systems is similar to that of prior software mechanisms [7]. Moreover, recent work [40] shows that a datacenter infrastructure, in case of a power emergency such as battery recharging under node oversubscription, needs to regulate the power consumption footprint of the nodes within a time window in the order of minutes. Based on that, we consider a cap enforcement interval of seconds, as in the case of RVSCap, to be adequate and safe for the infrastructure.

5.3 RVSCap with Hybrid Power Capping Mechanisms

As discussed in Section 4, hybrid power capping approaches exploit the nominal DVFS points of the CPU (by directly controlling frequency) in combination with the placement of threads to cores. PUPiL [7], a state-of-the-art work in CPU power capping, performs thread packing on top of RAPL to maximize performance. In this section, we evaluate the effects of having PUPiL use RVSCap instead of RAPL. On the X-Gene processors, we apply thread-packing on top of DFSCap and RVSCap. On the Intel CPU, we apply thread-packing on top of RAPL (as in [7]) and RVSCap, using RVSCap as a drop-in replacement for RAPL. Note that there is no explicit synergy between PUPiL and either RAPL or

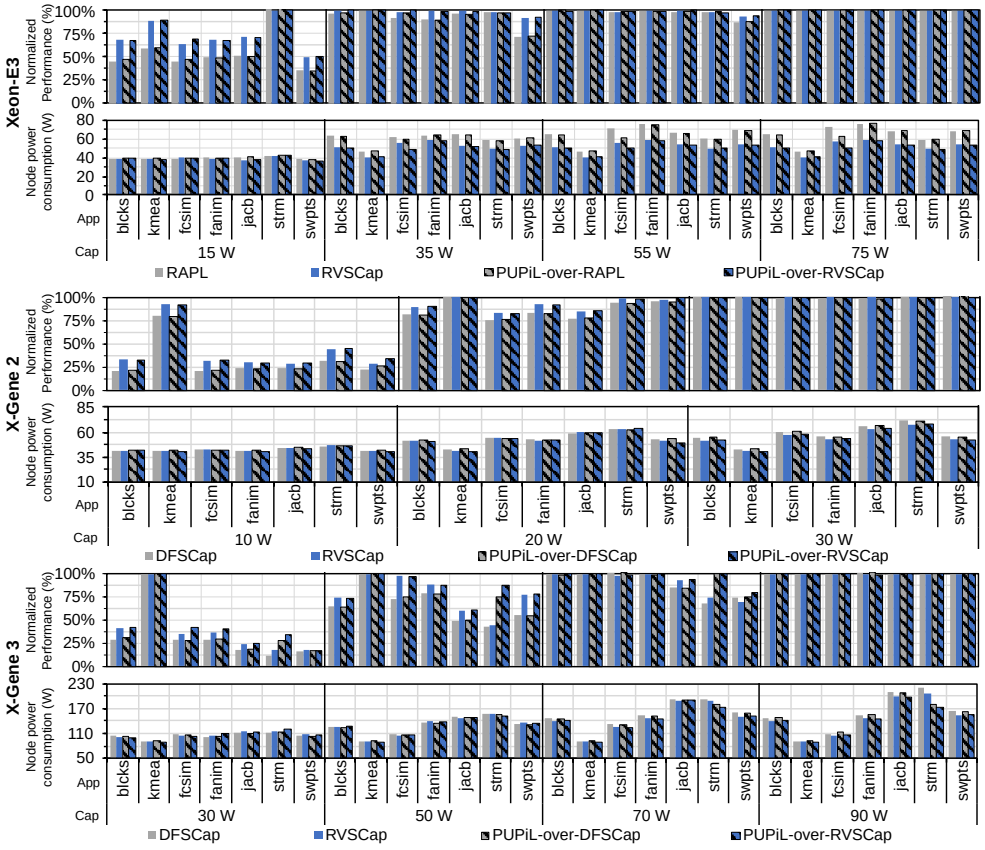

Fig. 10: Normalized performance (wrt. to uncapped execution) and node power consumption for RAPL, RVSCap, PUPiL-over-RAPL, and PUPiL-over-RVSCap on Xeon E3 and for DFSCap, RVSCap, PUPiL-over-DFSCap and PUPiL-over-RVSCap on X-Gene processors.

RVSCap. More specifically, PUPiL requests a power cap (to either RAPL, DFSCap, or RVSCap). RAPL, DFSCap, and RVSCap dynamically control CPU frequency and voltage to enforce the cap, while PUPiL independently controls CPU core utilization (thread-packing).

Figure 10 shows the performance and node power consumption for RAPL, RVSCap, PUPiL-over-RAPL, and PUPiL-over-RVSCap. Given that PUPiL [7] is applicable only to multithreaded workloads (where thread-packing can be performed), we use three additional multithreaded benchmarks: *k-means*, *jacobi* and *stream*, which were also not included in the characterization process of the CPUs presented in Section 3.2. To facilitate a more direct comparison with [7], we measure the performance as the rate of heartbeats that are delivered within a time window (*PUPiL employs a heartbeat framework implemented at the code level of the workload [38]*).

On Xeon E3, PUPiL-over-RAPL roughly matches RAPL in terms of performance, except for *facesim* where it outperforms RAPL. For restrictive power caps, both RVSCap and PUPiL-over-RVSCap outperform PUPiL-over-RAPL on average by 24% and 36%, respectively. This clearly shows that reduced voltage margins can provide additional benefits on top of state-of-the-art hybrid power capping methods. The limited effectiveness of PUPiL-over-RAPL, an average 1.5% performance improvement of PUPiL-over-RAPL vs RAPL, can be attributed to the fact that Xeon E3 is a single socket system and has only 4 CPU cores, whereas [7] evaluates PUPiL over RAPL on a dual-socket system with a total of 32 CPU cores.

The results are similar for X-Gene 2: 2.6% performance improvement, on average, of PUPiL-over-DFSCap vs DFSCap. The small number of CPU cores limits the performance gains of PUPiL-over-DFSCap, which is outperformed by RVSCap and PUPiL-over-RVSCap on average by 22% and 33%, respectively, for restrictive power caps. However, on X-Gene 3, the high CPU core count allows PUPiL-over-DFSCap to achieve greater performance gains for several benchmarks (such as *facesim* and *stream*), with an average 21% performance improvement vs DFSCap. Still, RVSCap and PUPiL-over-RVSCap continue to outperform PUPiL-over-DFSCap by 5% and 27% on average, respectively. The only case where PUPiL-over-DFSCap outperforms RVSCap is *stream*. Even in this case, PUPiL-over-RVSCap is the most efficient approach as it takes advantage of both thread-packing and the operation of the CPU at reduced voltage margins.

Our investigation shows that, as expected, thread-packing requires a high CPU cores count to achieve notable performance gains. In contrast, the reduction of voltage margins improves both performance and power efficiency irrespectively of the number of CPU cores. Also, the synergistic operation at reduced voltage margins with thread-packing (PUPiL-over-RVSCap) is not only possible but also leads to better results than each of these mechanisms separately.

5.4 Platforms Comparison

Based on these results, it is evident that the educated reduction of conservative CPU voltage margins significantly reduces the performance penalty introduced by conventional power capping approaches. Moreover, when the power consumption footprint of the workload remains below the power cap, the same performance is achieved with less power consumption.

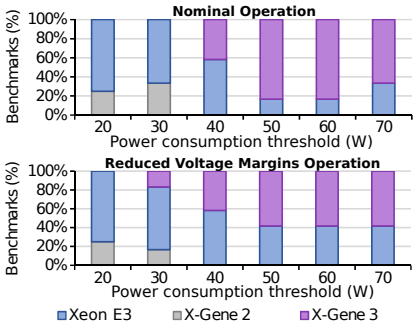

Fig. 11: Percentage of experiments (benchmarks) for which each processor achieved the highest throughput at a given power cap.

Recent works [1], [41], [42], [43], [44], [45] try to model and simulate different aspects of datacenters. They focus on maximizing the energy-efficiency and minimizing the costs for operators. Motivated by this line of research, we perform a comparison among the processors at hand, in terms of performance/throughput, to highlight the impact of voltage margin reduction on those aspects of datacenters.

More specifically, as shown in Figure 11, we quantify the percentage of benchmarks where each CPU proved to be the most efficient in terms of throughput under a specific power cap. We exclude X-Gene 2 from comparisons for power caps higher than the 30 Watt, as those caps exceed its TDP, therefore the comparison would be unfair.

For the conventional power capping mechanisms such as RAPL and DFSCap, as shown in Figure 11, Intel Xeon E3 is the most efficient processor, up to the 40 Watt power cap. For higher power envelopes, X-Gene 3 dominates, in terms of throughput. However, when reducing CPU voltage margins with the utilization of RVSCap (see Figure 11) Intel Xeon E3 becomes significantly more efficient, approaching the performance of X-Gene 3 for higher power consumption budgets. This is due to the fact that Intel Xeon E3 has wider voltage margins than X-Gene 3, which results in greater performance improvement under power-constrained execution.

Despite the narrower voltage margins of X-Gene 2, when the power budget is limited and lies below 20 Watts, as is often, for example, the case in Edge computing deployments, this is the only processor that is capable of operating within the designated budget. For higher power budgets, Intel Xeon E3 delivers sufficient throughput for cloud and single-threaded benchmarks. However, for benchmarks that are inherently parallel, X-Gene 3 is the most efficient CPU due to its higher core count.

6 Energy Gains under Checkpointing

The reduction of voltage margins, even when performed in an educated and careful manner, may affect the resilience of the CPU against errors. This would, in turn, make the system more prone to failures, reducing the Mean Time

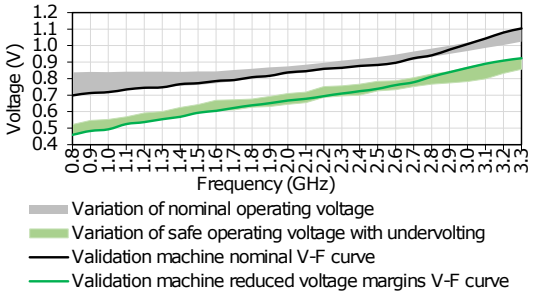

Fig. 12: Variation of CPU operating voltage at nominal and reduced voltage margins across the 16 Xeon E3 systems.

Between Failures (MTBF). In order to assess the risk of CPU operation at reduced voltage margins, we performed a long experimental evaluation on 16 identical Xeon E3 systems.

As a first step, we performed a characterization of voltage margins on all systems. Figure 12, illustrates the significant variation due to manufacturing variability across different parts (CPUs) of the same family, in terms of operating voltage, for both nominal (gray area) and at reduced voltage margins (green area) CPU operation. The black and green lines correspond to the nominal and reduced margins $(voltage, frequency)$ operating points of the system used for the experimental evaluation in the previous sections. The specific CPU part has an average voltage margin across all operating points at the median of the respective voltage margins quantified on all parts. The characterization confirms that the effects of manufacturing variability on the physical parameters of the chips are non-negligible and need to be taken into account in real-world scenarios.

During this long experimental campaign, RVSCap was constantly enabled on all systems, and random power caps in the range $[15W, 75W]$ (80W is the TDP of this CPU) were enforced on randomly selected combinations of applications from our benchmark set. The experiment was concluded after a total of 8832 device-hours, 552 hours (23 days) per system, without any observed failure such as crash, silent data corruption (SDC), or any detected error. To provide a pessimistic estimation on the MTBF of systems that operate with RVSCap, we assume that 1 system out of the 16 experiences a transient error at 552 hours.

Prior studies show that the MTBF of individual nodes in large-scale facilities can be captured by a Weibull distribution with decreasing hazard rate (DHR, shape parameter $\kappa = 0.7$) [46], [47], [48], [49]. In particular, we use the following equation

$$\text{MTBF} = \frac{1}{\lambda}\Gamma(1+\frac{1}{\kappa}) \tag{1}$$

where Γ refers to the gamma distribution, λ and κ are the scale parameter and the shape parameter of the Weibull distribution law, respectively. When $\kappa = 1$ the distribution is exponential with a constant failure rate and when $\kappa = 0.7$ the distribution transforms to DHR. We note that Weibull with DHR takes into account infant mortality phenomena, which is consistent with our own observations during the characterization process of the Xeon E3 CPUs where the selection of CPU voltages below V_{min} led to failures almost instantly.

In order to calculate the scale parameter (λ), we solve the following Weibull Cumulative Distribution Function (CDF) equation for λ

$$\text{F(t)} = 1 - e^{-(\frac{t}{\lambda})^{\kappa}} \tag{2}$$

where, based on the results of our experimental analysis, t equals to 552 hours, κ (shape parameter) equals to 0.7 and $F(t = 552\,\text{hours})$ equals to 6.25% because we assumed 1 system failing out of the total of 16 systems.

Furthermore, given the limited number of systems tested, we apply the chi-squared distribution to extrapolate and calculate the most pessimistic (minimum) MTBF for different confidence levels (CLs) [50], [51] based on the following equation

$$\text{MTBF}_{\text{CL}} = \frac{2\ \text{MTBF}}{X^2_{1-CL}(2n+2)} \tag{3}$$

where X^2 refers to the chi-square distribution, MTBF is the experimentally observed MTBF and n the number of failures. To this end, we assume a distribution with 2 degrees of freedom ($n = 0$) [50], [51], since we did not observe any errors during the testing. Eventually, based on the results of our risk assessment analysis, for a confidence level of 90% $MTBF_{0.9} \geq 634.79$ days. Note that it is common for industry assessments to provide MTBF at a 60% CL [52] which in our case is $MTBF_{0.6} \geq 1595.65$ days.

Assuming a platform that employs N nodes to execute a workload, the respective $MTBF_{plat} = MTBF_{node}/N$, where $MTBF_{plat}$ and $MTBF_{node}$ are the MTBF values for the whole platform and the node respectively, for any continuous failure distribution [46], as is the case with DHR Weibull. To deal with the increased probability of failure, large-scale systems typically perform checkpointing. Given that such mechanisms introduce overhead, we evaluate whether energy savings achieved thanks to operation at reduced voltage margins outweigh this overhead. Although there are more sophisticated checkpointing mechanisms [46], we pessimistically assume a blocking, coordinated checkpointing scheme that is performed at the optimal period to capture the impact of voltage margins reduction even in such inefficient implementations. More specifically, we use the following equations of the analysis in [46] which estimate the overhead (WASTE) and the optimal checkpointing period (T_{opt})

$$\text{WASTE} = 1 - \left(1 - \frac{C}{T}\right)\left(1 - \frac{1}{\text{MTBF}_{plat}}\left(D + R + \frac{T}{2}\right)\right) \tag{4}$$

$$T_{opt} = \sqrt{2(\text{MTBF}_{plat} - D - R)C} \tag{5}$$

where C is the checkpointing cost, T is the period of checkpointing, R is the cost of restoring the checkpoint, D is the downtime.

Figure 13 illustrates the energy benefits to be expected for a power capped execution at reduced margins with checkpointing vs. an execution at nominal CPU settings and no checkpointing, despite that in large-scales, nodes do fail and checkpointing schemes are necessary even when the CPU operates at nominal settings. Without loss of generality, we choose a constant $D = 2$ minutes and we let $C = R$ vary from 5 up to 60 minutes. The dashed black line represents

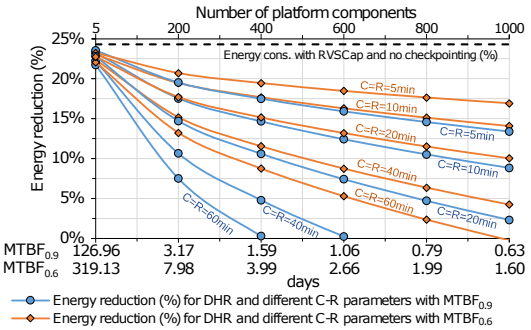

Fig. 13: Energy reduction of RVSCap with checkpointing vs. execution at nominal settings without checkpointing, for $MTBF_{0.9}$, $MTBF_{0.6}$ and different C-R parameters.

the energy gains when operating at reduced margins with the most relaxed power cap but without any checkpointing and assuming no failures (infinite MTBF). We do not show a comparison for more restrictive CPU power caps, as these anyway result in higher energy gains (see Section 5). Our analysis shows that RVSCap (and operation at reduced margins in general) can provide energy gains even at scale and for rather costly checkpointing implementations. For instance, in a system with 1000 nodes, operation at reduced margins remains beneficial if C, R is under 40 minutes for $MTBF_{0.6}$. For the much higher confidence level of $MTBF_{0.9}$, operation at reduced margins is beneficial as long as C, R is less than 20 minutes.

7 Related Work

Several approaches relax voltage guardbands in order to increase energy efficiency. In particular, [14], [15] present heuristics that dynamically reduce voltage margins, based on the feedback of error correction ECC mechanism built in Itanium 9560, and [16] rely on dedicated hardware protection mechanisms introduced at chip design-time to reduce voltage margins. In addition, [12], [13], [17], [18], [19], [20] present, similar to this work, static offline analysis that quantifies the CPU voltage margins on different commercial chips. Moreover, in contrast with this work, they are completely agnostic of CPU power capping, target CPU architectures that do not support hardware power capping, focus on a limited range of CPU frequencies and rely on specific CPU design, topology and behavior to carefully exploit the reduction of voltage margins. All these attributes make prior work incompatible with existing power capping approaches. Furthermore, none of the previous research efforts projects the potential energy gains in large-scale deployments, as this work does, where CPU power capping is typically useful, however, a fault tolerance mechanism, such as checkpointing, is also necessary.

Earlier power capping solutions [3], [27] are software-based, rely on external power meters and control CPU key factors such as frequency. However, recent works for improving performance under power-constrained environments, compare many existing power capping mechanisms [7], [53] for a wide range of scenarios, and report that RAPL capping almost ubiquitously outperforms previous software-based capping methods. Based on this observation, several state-of-the-art works propose hybrid solutions, which combine software mechanisms with RAPL.

In particular, [7] proposes PUPiL, a hybrid software-hardware framework, which uses RAPL in conjunction with a heuristic algorithm for identifying the best thread placement on cores scheme (thread-packing). [8] also relies on RAPL to cap the power of the CPU and RAM. It explores how cross-component power allocation of a single node will affect the performance of multi-threaded workloads, under a constrained power budget. [6] presents a holistic methodology that utilizes the techniques used in [7], [8] synergistically to further improve the performance of parallel workloads.

The previous works on power capping propose software-level optimizations, potentially in conjunction with hardware techniques such as RAPL, to improve performance in power-constrained environments. However, as shown in our investigation, CPU voltage margins reduction in the form of RVSCap results in better performance compared to the hybrid PUPiL over RAPL mechanism utilized in these works. Most importantly as shown, our approach is compatible with the above state-of-the-art solutions and could be employed in conjunction with them to provide additional performance benefits.

8 Conclusions

Our study investigates the impact of exploiting CPU voltage margins during power-constrained execution. Our investigation includes three different commercial CPUs, an Intel Xeon E3 x86-64 and the ARMv8 64-bit AppliedMicro X-Gene 2 and X-Gene 3. We experimentally show that CPU voltage margins reduction has a significant impact on the performance during power-constrained execution. More specifically, we observe that for restrictive power caps, CPU power capping with reduced voltage margins (RVSCap) improves the performance by 64% for Xeon E3 (RAPL), 30% for X-Gene 2 (DFSCap) and 34% for X-Gene 3 (DFSCap), on average. Also for less restrictive power caps, the same performance with the conventional capping mechanisms is met, but with minimized node power footprint which is on average by 24%, 4% and 5% lower for Xeon E3 (RAPL), X-Gene 2 (DFSCap) and X-Gene 3 (DFSCap), respectively.

In addition, CPU voltage margins reduction leads to greater performance gains even when compared with the state-of-the-art solution in CPU power capping (PUPiL) by 24%, 22% and 5% on average for Xeon E3, X-Gene 2 and X-Gene 3, respectively. Noticeably, we show that the synergy between these is not only possible but also outperforms every other approach, in terms of performance.

More importantly, our investigation includes long validation runs and a risk assessment analysis which estimates the MTBF of Xeon E3 CPUs, when operating with reduced voltage margins, and proves that CPU voltage margins reduction can result to energy gains even at larger scales and even when considering the increased checkpointing overhead, assuming a potentially reduced system MTBF.

Acknowledgments

This work is funded by the H2020 Framework Program of the European Union through the UniServer Project (Grant

Agreement 688540) – http://www.uniserver2020.eu.

REFERENCES

[1] J. Koomey, "Growth in Data Center Electricity Use 2005 to 2010," *A report by Analytical Press, completed at the request of The New York Times*, vol. 9, 2011.

[2] M. Dayarathna, Y. Wen, and R. Fan, "Data Center Energy Consumption Modeling: A Survey," *Journal of IEEE Communications Surveys & Tutorials*, vol. 18, no. 1, pp. 732–794, 2016.

[3] C. Lefurgy, X. Wang, and M. Ware, "Power Capping: A Prelude to Power Shifting," *Journal of Cluster Computing*, vol. 11, no. 2, pp. 183–195, Jun. 2008.

[4] X. Wang and M. Chen, "Cluster-level Feedback Power Control for Performance Optimization," in *Proceedings of the IEEE 14th International Symposium on High Performance Computer Architecture (HPCA)*, Feb 2008, pp. 101–110.

[5] R. Cochran, C. Hankendi, A. K. Coskun, and S. Reda, "Pack & Cap: Adaptive DVFS and Thread Packing Under Power Caps," in *Proceedings of the 44th Annual IEEE/ACM International Symposium on Microarchitecture (MICRO)*, Dec 2011, pp. 175–185.

[6] J. Park, S. Park, and W. Baek, "RPPC: A Holistic Runtime System for Maximizing Performance Under Power Capping," in *Proceedings of the 18th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (CCGRID)*. IEEE, 2018, pp. 41–50.

[7] H. Zhang and H. Hoffmann, "Maximizing Performance Under a Power Cap: A Comparison of Hardware, Software, and Hybrid techniques," *Journal of ACM SIGARCH Computer Architecture News*, vol. 44, no. 2, pp. 545–559, 2016.

[8] R. Ge, X. Feng, Y. He, and P. Zou, "The Case for Cross-Component Power Coordination on Power Bounded Systems," in *Proceedings of the 45th International Conference on Parallel Processing (ICPP)*. IEEE, 2016, pp. 516–525.

[9] F. Sun, H. Li, Y. Han, G. Yan, and J. Ma, "PowerCap: Leverage Performance-Equivalent Resource Configurations for Power Capping," in *Proceedings of the 7th International Green and Sustainable Computing Conference (IGSC)*. IEEE, 2016, pp. 1–8.

[10] A. Beloglazov, J. Abawajy, and R. Buyya, "Energy-Aware Resource Allocation Heuristics for Efficient Management of Data Centers for Cloud Computing," *Journal of Future Generation Computer Systems*, vol. 28, no. 5, pp. 755–768, 2012.

[11] H. David, E. Gorbatov, U. R. Hanebutte, R. Khanna, and C. Le, "RAPL: Memory Power Estimation and Capping," in *Proceedings of the 16th ACM/IEEE International Symposium on Low Power Electronics and Design (ISLPED)*. ACM, 2010, pp. 189–194.

[12] K. Parasyris, P. Koutsovasilis, V. Vassiliadis, C. D. Antonopoulos, N. Bellas, and S. Lalis, "A Framework for Evaluating Software on Reduced Margins Hardware," in *Proceedings of the 48th Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN)*. IEEE, 2018, pp. 330–337.

[13] G. Papadimitriou, M. Kaliorakis, A. Chatzidimitriou, D. Gizopoulos, P. Lawthers, and S. Das, "Harnessing Voltage Margins for Energy Efficiency in Multicore CPUs," in *Proceedings of the 50th Annual IEEE/ACM International Symposium on Microarchitecture (MICRO)*. ACM, 2017, pp. 503–516.

[14] A. Bacha and R. Teodorescu, "Dynamic Reduction of Voltage Margins by Leveraging on-chip ECC in Itanium II Processors," in *Proceedings of the ACM SIGARCH Computer Architecture News*, vol. 41, no. 3. ACM, 2013, pp. 297–307.

[15] A. Bacha and R. Teodorescu, "Using ECC Feedback to Guide Voltage Speculation in Low-Voltage Processors," in *Proceedings of the 47th Annual IEEE/ACM International Symposium on Microarchitecture (MICRO)*. IEEE Computer Society, 2014, pp. 306–318.

[16] Y. Zu, C. R. Lefurgy, J. Leng, M. Halpern, M. S. Floyd, and V. J. Reddi, "Adaptive Guardband Scheduling to Improve System-level Efficiency of the POWER7+," in *Proceedings of the 48th Annual IEEE/ACM International Symposium on Microarchitecture (MICRO)*. IEEE, 2015, pp. 308–321.

[17] G. Papadimitriou, A. Chatzidimitriou, and D. Gizopoulos, "Adaptive Voltage/Frequency Scaling and Core Allocation for Balanced Energy and Performance on Multicore CPUs," in *Proceedings of the IEEE International Symposium on High Performance Computer Architecture (HPCA)*. IEEE, 2019, pp. 133–146.

[18] K. Tovletoglou, L. Mukhanov, G. Karakonstantis, A. Chatzidimitriou, G. Papadimitriou, M. Kaliorakis, D. Gizopoulos, Z. Hadjilambrou, Y. Sazeides, A. Lampropulos, S. Das, and P. Vo, "Measuring and Exploiting Guardbands of Server-Grade ARMv8 CPU Cores and DRAMs," in *Proceedings of the 48th Annual IEEE/IFIP International Conference on Dependable Systems and Networks Workshops (DSN-W)*. IEEE, jun 2018.

[19] G. Papadimitriou, A. Chatzidimitriou, M. Kaliorakis, Y. Vastakis, and D. Gizopoulos, "Micro-Viruses for Fast System-Level Voltage Margins Characterization in Multicore CPUs," in *Proceedings of the IEEE International Symposium on Performance Analysis of Systems and Software (ISPASS)*, April 2018, pp. 54–63.

[20] G. Papadimitriou, M. Kaliorakis, A. Chatzidimitriou, C. Magdalinos, and D. Gizopoulos, "Voltage Margins Identification on Commercial x86-64 Multicore Microprocessors," in *Proceedings of the IEEE 23rd International Symposium on On-Line Testing and Robust System Design (IOLTS)*. IEEE, jul 2017.

[21] M. Ferdman, A. Adileh, O. Kocberber, S. Volos, M. Alisafaee, D. Jevdjic, C. Kaynak, A. D. Popescu, A. Ailamaki, and B. Falsafi, "Clearing the Clouds: a Study of Emerging Scale-out Workloads on Modern Hardware," in *Proceedings of the ACM SIGPLAN Notices*, vol. 47, no. 4. ACM, 2012, pp. 37–48.

[22] C. Bienia, S. Kumar, J. P. Singh, and K. Li, "The PARSEC Benchmark Suite: Characterization and Architectural Implications," in *Proceedings of the 17th International Conference on Parallel Architectures and Compilation Techniques (PACT)*. ACM, 2008, pp. 72–81.

[23] J. L. Henning, "SPEC CPU2006 Benchmark Descriptions," *Journal of ACM SIGARCH Computer Architecture News*, vol. 34, no. 4, pp. 1–17, 2006.

[24] J. J. Dongarra, P. Luszczek, and A. Petitet, "The LINPACK Benchmark: Past, Present and Future," *Journal of Concurrency and Computation: Practice and Experience*, vol. 15, no. 9, pp. 803–820, 2003.

[25] G. Woltman, "Prime95," 2012. [Online]. Available: https://www.mersenne.org/download/

[26] D. Hackenberg, R. Oldenburg, D. Molka, and R. Schöne, "Introducing FIRESTARTER: A Processor Stress Test Utility," in *Proceedings of the International Green Computing Conference (IGCC)*. IEEE, 2013, pp. 1–9.

[27] X. Wang and M. Chen, "Cluster-level Feedback Power Control for Performance Optimization," in *Proceedings of the IEEE 14th International Symposium on High Performance Computer Architecture (HPCA)*, Feb 2008, pp. 101–110.

[28] H. Sasaki, A. Buyuktosunoglu, A. Vega, and P. Bose, "Characterization and Mitigation of Power Contention Across Multiprogrammed Workloads," in *Proceedings of the IEEE International Symposium on Workload Characterization (IISWC)*. IEEE, sep 2016.

[29] AppliedMicro (APM), "APM883832-X3 — X-Gene 3© Multi-Core 64-bit Processor," 2016, https://en.wikichip.org/w/images/2/22/832-X3_PB.pdf.

[30] R. J. Wysocki, "ACPI 6 and Linux," 2015. [Online]. Available: http://events17.linuxfoundation.org/sites/events/files/slides/ACPI_6_and_Linux_0.pdf

[31] E. A. Burton, G. Schrom, F. Paillet, J. Douglas, W. J. Lambert, K. Radhakrishnan, and M. J. Hill, "FIVR—Fully Integrated Voltage Regulators on 4th Generation Intel® Core ™ SoCs," in *Proceedings of 29th Annual International Conference on Applied Power Electronics Conference and Exposition (APEC)*. IEEE, 2014, pp. 432–439.

[32] J. Doweck, W.-F. Kao, A. K.-y. Lu, J. Mandelblat, A. Rahatekar, L. Rappoport, E. Rotem, A. Yasin, and A. Yoaz, "Inside 6th-Generation Intel Core: new Microarchitecture Code-named Skylake," *Journal of IEEE Micro*, no. 2, pp. 52–62, 2017.

[33] E. Cai and D. Marculescu, "TEI-Turbo: Temperature Effect Inversion-Aware Turbo Boost for Finfet-Based Multi-Core Systems," in *Proceedings of the IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, Nov 2015, pp. 500–507.

[34] W. Lee, Y. Wang, T. Cui, S. Nazarian, and M. Pedram, "Dynamic Thermal Management for FinFET-Based Circuits Exploiting the Temperature Effect Inversion Phenomenon," in *Proceedings of the IEEE/ACM International Symposium on Low Power Electronics and Design (ISLPED)*, Aug 2014, pp. 105–110.

[35] S. Sankar, M. Shaw, K. Vaid, and S. Gurumurthi, "Datacenter Scale Evaluation of the Impact of Temperature on Hard Disk Drive Failures," *Journal of ACM Transactions on Storage (TOS)*, vol. 9, no. 2, p. 6, 2013.

[36] M. Patel, J. S. Kim, H. Hassan, and O. Mutlu, "Understanding and Modeling On-Die Error Correction in Modern DRAM: An Experimental Study Using Real Devices," in *Proceedings of the 49th*

Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN). IEEE, 2019, pp. 13–25.
[37] A. T. Committee, "Data Center Power Equipment Thermal Guidelines and Best Practices Whitepaper," June 2016, https://tc0909.ashraetcs.org/documents/ASHRAE_TC0909_Power_White_Paper_22_June_2016_REVISED.pdf.
[38] H. Hoffmann, J. Eastep, M. D. Santambrogio, J. E. Miller, and A. Agarwal, "Application Heartbeats: a Generic Interface for Specifying Program Performance and Goals in Autonomous Computing Environments," in *Proceedings of the 7th International Conference on Autonomic Computing (ICAC)*. ACM, 2010, pp. 79–88.
[39] J. Marusarz, S. Cepeda, and A. Yasin, "How to Tune Applications Using a Top-Down Characterization of Microarchitectural Issues," in *Technical report*. Intel, 2013.
[40] S. Malla, Q. Deng, Z. Ebrahimzadeh, J. Gasperetti, S. Jain, P. Kondety, T. Ortiz, and D. Vieira, "Coordinated Priority-aware Charging of Distributed Batteries in Oversubscribed Data Centers," in *Proceedings of the 53rd Annual IEEE/ACM International Symposium on Microarchitecture (MICRO)*, 2020.
[41] C. Kalogirou, P. Koutsovasilis, C. D. Antonopoulos, N. Bellas, S. Lalis, S. Venugopal, and C. Pinto, "Exploiting CPU Voltage Margins to Increase the Profit of Cloud Infrastructure Providers," in *Proceedings of the 19th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (CCGRID)*. IEEE, 2019, pp. 302–311.
[42] A. Ouammou, A. B. Tahar, M. Hanini, and S. El Kafhali, "Modeling and Analysis of Quality of Service and Energy Consumption in Cloud Environment," *International Journal of Computer information Systems and Industrial Management Applications*, vol. 10, pp. 98–106, 2018.
[43] A. Ouammou, M. Hanini, S. El Kafhali, and A. B. Tahar, "Energy Consumption and Cost Analysis for Data Centers with Workload Control," in *Proceedings of the International Conference on Innovations in Bio-Inspired Computing and Applications*. Springer, 2017, pp. 92–101.
[44] M. Wardat, M. Al-Ayyoub, Y. Jararweh, and A. A. Khreishah, "Cloud Data Centers Revenue Maximization Using Server Consolidation: Modeling and Evaluation," in *Proceedings of the IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*. IEEE, 2018, pp. 172–177.
[45] S. S. Gill and R. Buyya, "A Taxonomy and Future Directions for Sustainable Cloud Computing: 360 Degree View," *arXiv preprint arXiv:1712.02899*, 2017.
[46] J. Dongarra, T. Herault, and Y. Robert, "Fault Tolerance Techniques for High-Performance Computing," in *Fault-Tolerance Techniques for High-Performance Computing*. Springer, 2015, pp. 3–85.
[47] T. J. Hacker, F. Romero, and C. D. Carothers, "An Analysis of Clustered Failures on Large Supercomputing Systems," *Journal of Parallel and Distributed Computing*, vol. 69, no. 7, pp. 652–665, 2009.
[48] Y. Liu, R. Nassar, C. Leangsuksun, N. Naksinehaboon, M. Paun, and S. L. Scott, "An Optimal Checkpoint/Restart Model for a Large Scale High Performance Computing System," in *Proceedings of the IEEE International Symposium on Parallel and Distributed Processing (IPDPS)*. IEEE, 2008, pp. 1–9.
[49] B. Schroeder and G. A. Gibson, "A Large-Scale Study of Failures in High-Performance Computing Systems," in *Proceedings of the International Conference on Dependable Systems and Networks (DSN)*, June 2006, pp. 249–258.
[50] MIL-HDBK-338, "Military Handbook. Electronic Reliability Design Handbook," 1998. [Online]. Available: https://www.weibull.com/knowledge/milhdbk.htm
[51] I. Bazovsky, *Reliability Theory and Practice*. Courier Corporation, 2004.
[52] Intel, "Reliability report," 1H 2017, https://www.intel.com/content/dam/www/programmable/us/en/pdfs/literature/rr/rr.pdf.
[53] P. Petoumenos, L. Mukhanov, Z. Wang, H. Leather, and D. S. Nikolopoulos, "Power Capping: What Works, What Does Not," in *Proceedings of the 21st International Conference on Parallel and Distributed Systems (ICPADS)*. IEEE Computer Society, 2015, pp. 525–534.

Panos Koutsovasilis is a PhD candidate at the ECE Department of the University of Thessaly in Volos, Greece. His research focuses on system level programming (operating systems, schedulers, execution policies), virtualization, energy efficiency and reliability on heterogeneous systems. He is also interested on high performance computing, parallel programming and software optimizations. He received his MSc in computer engineering from the ECE Department of the University of Thessaly, Greece.

Christos D. Antonopoulos is an Assistant Professor at the ECE Department of the University of Thessaly in Volos, Greece. His research interests span the areas of system and applications software for high performance computing, emphasizing on run-time monitoring and adaptivity with performance and power criteria. He also works on improving the programmability of accelerator-based heterogeneous systems, as well as on techniques for automatic, application-driven redefinition of the hardware/software boundary on accelerator-based systems. He earned his PhD, MSc and Diploma from the Computer Engineering and Informatics Department of the University of Patras, Greece.

Nikolaos Bellas is an Associate Professor at the ECE Department of the University of Thessaly in Volos, Greece. His research interests include CAD tools for architectural synthesis, reliable and low power computing, embedded systems, and reconfigurable computing. He received his Diploma from the Computer Engineering and Informatics Department of the University of Patras, Greece, and the M.Sc. and Ph.D. degrees in Electrical and Computer Engineering from the University of Illinois, Urbana-Champaign.

Spyros Lalis is an Associate Professor at the ECE Department of the University of Thessaly in Volos, Greece. His research interests are mainly in programming models, operating systems, distributed systems and ubiquitous computing systems. He has contributed to the development of system-level software, middleware and programming frameworks to support market-based resource allocation, distributed computing, ad-hoc wearable and personal computing, wireless sensor networks and approximate computing. He received a Diploma in Computer Science and a PhD in Technical Sciences from the Swiss Federal Institute of Technology Zurich (ETHZ).

George Papadimitriou is a Postdoctoral Researcher at the Dept. of Informatics & Telecommunications of the University of Athens. He graduated from the Dept. of Computer Systems Engineering of the Technological Educational Institute of Piraeus as valedictorian, where he received his BSc degree in Computer Engineering in 2011. He received his MSc with honours in Computer Systems Technology and his PhD in Computer Architecture from the Dept. of Informatics & Telecommunications of the University of Athens. His research focuses on dependability and energy-efficient computer architectures, microprocessor reliability, functional correctness of hardware designs and design validation of microprocessors and microprocessor-based systems, in which he has published more than 20 papers in international conferences and journals. He has co-organized and participated in several tutorials around these research areas including MICRO 2018, and ISCA 2017 / 2018, and has been distinguished two times with the HiPEAC paper award for his publications in the two most competitive computer architecture conferences (MICRO 2017 and HPCA 2019).

Athanasios Chatzidimitriou received his B.Sc. degree on Informatics engineering from Technological Educational Institute of Athens, Greece, his M.Sc. degree on Embedded Computing Systems from University of Piraeus, Greece and his Ph.D. on Computer Architecture from Department of Informatics and Telecommunications at University of Athens. His research interests focus on microprocessor reliability where he has published more than 25 papers in international conferences and journals.

Dimitris Gizopoulos is Professor at the Department of Informatics & Telecommunications, of National and Kapodistrian University of Athens where he leads the Computer Architecture Laboratory. His research focuses on the dependability, correctness, performance and power efficiency of computing systems architectures built around high-performance and embedded multicore CPUs as well as GPUs and other accelerators. Gizopoulos has published more than 180 papers in peer reviewed IEEE and ACM journals and conferences, has received best awards and awards nominations, is author of one book and editor of three more books on dependable computing. He currently serves as Associate Editor for IEEE Transactions on Computers (TC), IEEE Transactions on Sustainable Computing (TSUSC), IEEE Transactions on Emerging Topics in Computing (TETC) and Springer's Journal of Electronic Testing. In the past, he served as Associate Editor for several IEEE Transactions and Magazines and guest editor in special issues for several IEEE Transactions and Magazines and as Program Chair, General Chair and member of the Steering, Organizing and Program Committees of major IEEE and ACM conferences as well as on several IEEE Awards and Fellowships evaluation committees. Gizopoulos is an IEEE Fellow, a Golden Core Member of IEEE Computer Society, and an ACM Senior Member