CoolBudget: Data Center Power Budgeting with Workload and Cooling Asymmetry Awareness

Ozan Tuncer*, Kalyan Vaidyanathan†, Kenny Gross†, Ayse K.Coskun*
*ECE Department, Boston University, Boston, MA 02215, USA, {otuncer, acoskun}@bu.edu
†Oracle Physical Sciences Research Center, San Diego, CA 92121, USA, {kalyan.vaidyanathan, kenny.gross}@oracle.com

***Abstract*—Power over-subscription challenges and emerging cost management strategies motivate designing efficient data center power capping techniques. During capping, provisioned power must be budgeted among the computational and cooling units. This work presents a data center power budgeting policy that simultaneously improves the quality-of-service (QoS) and power efficiency by considering the workload- and cooling-induced asymmetries among the servers. Proposed policy finds the most efficient data center temperature and the power distribution among servers while guaranteeing reliable temperature levels for the server internal components. Experiments based on real servers demonstrate 21% increase in throughput compared to existing techniques.**

I. Introduction

Increasing power consumption has become a major concern in enterprise data centers due to its contribution to operational cost, reliability constraints and environmental concerns. To reduce the electricity costs, data centers employ power provisioning to increase the overall utilization [9], and more recently, contemplate participating in renewable energy use or demand-side power regulation programs [4]. These cost management mechanisms require the data centers to have the ability for *capping* their total power consumption.

Efficient power capping implies maximizing QoS under a total data center power budget. To achieve this goal, one has to distribute the available power budget across the servers in the data center, taking the performance demands of the applications into account. As data center cooling can consume over 30% of the total data center electricity [11], power budgeting should also account for the cooling power.

In this work, we propose a novel data center power budgeting technique to optimally distribute the power among cooling units and servers in a way that is aware of the thermal and workload asymmetries among the servers. Workload asymmetries arise because of the different power-performance characteristics of individual jobs, and thermal asymmetries occur because of both power differences of jobs and the heterogeneous *heat recirculation* effects in the data center. Our policy limits the power given to hotter servers to reduce the data center level cooling needs and, in this way, redirect more of the cooling power into computational power.

We also observe that many servers are over-cooled in today's data centers as the inlet temperature recommendations of manufacturers, which are based on worst-case conditions (i.e., utilization, altitude, etc.), leave a large thermal headroom margin (THM) between the server internal temperatures and critical thermal thresholds. Using empirical temperature and power models for the servers, our method safely collapses the THM to reduce the cooling power while maintaining safe temperatures. Our specific contributions are as follows:

- We propose a workload-aware data center power budgeting policy that finds the optimal room temperature and distributes the given budget among the cooling units and servers to maximize *fair speedup* of the jobs.
- Our policy reduces the headroom between the server internal temperatures and critical thresholds using accurate power, performance and temperature models that are validated based on measurements on a real enterprise server. Reducing the thermal headroom increases the data center throughput by 21% without introducing reliability concerns.
- We demonstrate the role of job allocation in conjunction with our proposed method. Simulations show that thermally-aware job allocation increases the throughput per Watt by up to 45% in comparison to the random job allocation.

II. Related Work

Existing techniques to reduce the energy use in data centers mainly focus on reducing the cooling need through thermally efficient job allocation [10], [14]. Only a few of such techniques address the budgeting problem to increase efficiency [6], [7]. Such techniques, however, do not budget the given power cap among cooling and computing, but solely focus on power control of the computational units.

A similar technique to ours is proposed by Zhan et al. [15]. Their solution allocates sufficient power for cooling and selects dynamic voltage-frequency scaling (DVFS) settings for each server to maximize throughput using a multi-choice knapsack algorithm. Their policy depends on computationally expensive computational fluid dynamics simulations. To the best of our knowledge, our work is the first to distribute the power optimally between cooling and computing. In addition, our policy uses server internal temperatures while determining the cooling temperature to improve efficiency.

III. Experimental Methodology

In order to analyze the relationships between server power, processor temperature, and throughput, we experiment on a presently-shipping enterprise server with two SPARC T3 CPUs in 2 sockets. Each CPU has 16 8-way hyperthreaded cores, providing a total of 256 simultaneous hardware threads. We collect sensor measurements of processor voltage, current, and temperature for every CPU, as well as total server power and server inlet temperature through Continuous System Telemetry Harness (CSTH) [8]. In addition, we collect performance counter data from the memory busses and the processors.

This work has been partially funded by Oracle Corp.

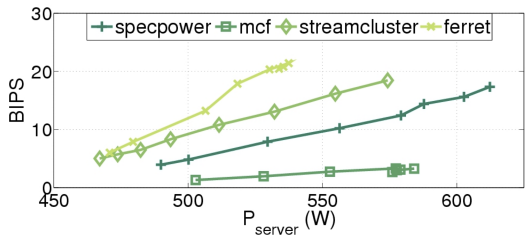

Fig. 1. BIPS vs. server power relationship for various jobs, when each job is running with 8, 12, 16, 20, 24, 28, 32 cores.

We control the server power consumption using *thread packing* [5], i.e., we allocate the software threads in a fewer number of hardware threads to decrease the power. We apply thread-packing at core level by either activating or deactivating all 8 threads in a core using Solaris `psrset` tool. The 32 cores in our server enables 32 possible power states. Finer-grained capping can also be implemented if desired (e.g. [4]).

In our experiments, we run SPECpower_ssj 2008 [13], PARSEC 2.1 [3], and a subset of the SPEC CPU2006 [12] that comprises different workload characteristics. Our total experimental database consists of 300 jobs, out of which, 75% are randomly selected for model training, and the remaining 25% are used for validation. In addition to the benchmarks, we use a custom-designed synthetic workload tool, *LoadGen*, to stress our server with any desired utilization level and to thoroughly model the temperature-power relationships.

Our target data center consists of 40 racks with 9 servers per rack. The racks are distributed in 4 rows with a cool-aisle hot-aisle configuration. Two computer room air conditioning units (CRACs) are located at the same side of the two hot aisles and use under-floor cooling with room return. We use *TileFlow* [2] to model the data center heat flow dynamics.

IV. Constructing Telemetry-based Models for Efficient Budgeting

In order to estimate performance, power, and temperature for each server under various power distribution scenarios, we develop empirical models based on telemetry collected from our enterprise server. Our modeling methodology is applicable to a wide range of server hardware and workload scenarios.

A. Server Power and Throughput

We use billions of instructions per second (BIPS) as the throughput metric, and observe a linear relationship between the server power cap, P_{server}, and BIPS as shown in Figure 1. We model BIPS as follows:

$$BIPS = k_0 \cdot P_{server} + k_1 \tag{1}$$

where k_0 and k_1 are constants that depend on the job. At runtime, we estimate $k_{0,1}$ using the model proposed by Zhan et al. [15] with performance counter data and power measurements. In our database, predicting BIPS at different P_{server} levels this way has a mean error of 0.6 BIPS (corresponding to only 3% average error) and a standard deviation of 2.5 BIPS.

Note that in Figure 1, the increase in throughput stops at a certain power level for each job. This is either due to a resource bottleneck (see *ferret* and *mcf*), or because increasing the number of active cores further than the number of software threads does not bring any benefits. To predict the maximum

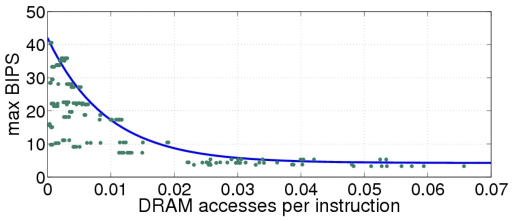

Fig. 2. BIPS upper-bound set by the number of DRAM accesses per instruction. The dots represent individual jobs and the solid line is the regression result for the upper-bound.

achievable BIPS and power levels by a given job, we first assume linear scaling of BIPS with the number of threads:

$$maxBIPS_{threads} = BIPS \cdot \frac{n_{sw\_threads}}{n_{hw\_threads}} \tag{2}$$

where $n_{sw\_threads}$ and $n_{hw\_threads}$ are the number of software and active hardware threads, respectively, and $BIPS$ is runtime measurement. Second, we observe that the number of DRAM accesses per instruction of an application puts an upper bound on the maximum BIPS achievable by a job as shown in Figure 2, where the data belongs to the jobs whose BIPS-power scaling is limited (e.g. due to a bottleneck). The upper bound on BIPS has the following form:

$$maxBIPS_{bottleneck} = k_2 + k_3 \cdot e^{k_4 \cdot n_{DRAM}} \tag{3}$$

where $k_{2,3,4}$ are regression coefficients. The maximum achievable BIPS while running a given job, $maxBIPS$, is the minimum of Equations 2 and 3. Given $maxBIPS$, maximum server power, $P^i_{server,max}$, is calculated using Equation 1. Predicting maximum server power in this way has a mean error of 11W and a standard deviation of 30W in our server, which consumes between 400-700W in our experiments.

B. Server Internal Temperatures

When assigning power limits to individual servers, we need to guarantee that the temperature thresholds are not violated. As the CPU is the hottest component in our server, we focus on the CPU temperature, T_{cpu}. The same methodology can be used for other server components such as GPUs.

We cap the power consumption of each server during budgeting; thus, temperature is also limited to the steady-state value achievable by the power cap. Based on the thermal RC model of the chip, the steady-state CPU temperature depends on its power consumption, P_{cpu}, the thermal resistance, R_{cpu}, which is determined by the hardware characteristics and the server fan speed, and the server inlet temperature:

$$T_{cpu} = R_{cpu} \cdot P_{cpu} + T_{inlet} + k_5 \tag{4}$$

where k_5 represents an empirical ΔT that reflects the increase in the air temperature within the server enclosure. We derive the thermal characteristics of our CPU using LoadGen at different utilization levels under a fixed fan speed of 2400 rpm, which is an empirically selected value that prevents the server leakage power to become dominant over the fan power.

As P_{cpu} and BIPS are generally correlated, we model P_{cpu} using the same methodology in BIPS estimation as follows:

$$P_{cpu} = k_6 \cdot P_{server} + k_7 \tag{5}$$

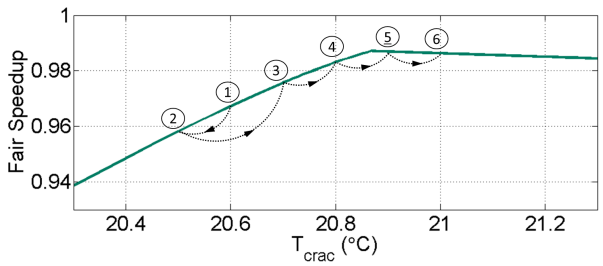

Fig. 3. Typical trend in maximum fair speedup within the proximity of optimum T_{crac}, and the policy iteration steps with a starting point of 20.6°C

where $k_{6,7}$ are calculated in the same way as $k_{0,1}$ in Equation 1. The combined CPU power-temperature model overpredicts the CPU temperature for a given server power with a mean of 2.9°C and a standard deviation of 1.5°C.

C. Data Center Thermal Dynamics

We model the server inlet temperatures using the methodology proposed by Tang et al. [14]. The inlet temperature of a server is represented by a linear combination of the CRAC outlet temperature and the power consumption of each server::

$$\boldsymbol{T_{inlet}} = \boldsymbol{D}\boldsymbol{P_{server}} + T_{crac} \tag{6}$$

where $\boldsymbol{T_{inlet}}$ and $\boldsymbol{P_{server}}$ are the server inlet temperature and power vectors, respectively, T_{crac} is the CRAC outlet temperature, and $\boldsymbol{D}$ is the heat distribution matrix. We calculate the heat distribution matrix of our target data center using thermal simulations with *TileFlow* [2].

We model the CRAC unit power consumption using the coefficient of performance (CoP) approach. CoP is defined as $CoP = P_{compute}/P_{cool}$, where $P_{compute}$ is the total computing power (all servers) and P_{cool} is the cooling power. We use the CoP model given by Moore et al. [10] as follows:

$$CoP = 0.0068 \cdot T_{crac}^2 + 0.0008 \cdot T_{crac} + 0.458 \tag{7}$$

V. Power Budgeting Policy

Our power budgeting policy aims to optimally distribute a given total power among the cooling and computing units, where the overall data center performance is maximized without an unfair performance degradation for any of the jobs.

A. CoolBudget Policy Overview

In order to do a workload-aware power budgeting, the policy first collects performance counter data from all servers and constructs the power and temperature models described in Section IV. Using these models, an optimization problem is iteratively solved to find the most efficient power distribution.

The policy maximizes the fair speedup, which corresponds to the harmonic mean of per server speedup, defined as:

$$\text{Fair Speedup} = \frac{N}{\sum_i^N (maxBIPS^i/BIPS^i)} \tag{8}$$

where N is the number of servers and $BIPS^i$ is the throughput of job i. Fair speedup is both an indicator of overall performance and a measure of fairness.

Our policy computes the optimum power distribution among the servers for a given T_{crac}. In order to find the most efficient T_{crac}, the problem is iteratively solved at different CRAC temperatures. Figure 3 shows the typical trend in maximum fair speedup with respect to T_{crac} for a given total budget. When T_{crac} is increasing, the fair speedup first increases because of the decrease in the cooling power due to Equation 7. This means that a larger portion of the total power budget is used for computation, leading to a higher throughput. When T_{crac} raises above a certain level, the performance of the hottest servers are degraded considerably to keep the temperature under the redline; thus, an increase in the room temperature is not useful anymore for the overall objective.

Based on the observation above, CoolBudget starts searching for the most efficient T_{crac} using its last known optimal value, which is 20.6°C in Figure 3. The policy first solves the optimization problem in the proximity of the last optimal T_{crac} (steps 1, 2, and 3 in the figure). Then, it iterates in the direction of increasing fair speedup (4 and 5) until fair speedup starts decreasing (6). Finally, the best solution is selected (5). We use 0.1°C resolution for the T_{crac} selection.

B. Optimization Problem

The optimization problem finds the best power distribution among the servers for a given T_{crac}, and formulated as:

$$\min_{\boldsymbol{P_{server}}} \quad \sum_i^N (maxBIPS^i/BIPS^i) \tag{9a}$$

$$\text{s.t.} \quad (1 + 1/CoP) \sum_i P^i_{server} \leq P_{budget} \tag{9b}$$

$$R_{cpu}(k_6 P^i_{server} + k_7) + (\boldsymbol{D}\boldsymbol{P_{server}})^i + T_{crac} + k_5 \leq T^i_{redline} \quad \forall i \tag{9c}$$

$$P^i_{server,idle} \leq P^i_{server} \leq P^i_{server,max} \quad \forall i \tag{9d}$$

The objective function in (9a) is the denominator of Equation 8. Constraint (9b) limits the total power usage, and is derived from the equation $P_{compute} + P_{cool} \leq P_{budget}$ and CoP equations (see Section IV-C). (9c) keeps the temperature of each processor under a given redline by combining Equations 4, 5, and 6. This constraint also introduces location-awareness to the problem through the heat recirculation matrix $\boldsymbol{D}$. Finally, (9d) ensures that the power given to a server falls between the idle power and maximum power for the given job.

CoolBudget is invoked every second. As the jobs we use generally have stable power profiles when they are executing, and because the thermal time constants of the CPUs in our server are in the order of tens of seconds, the periodic check of 1 second is sufficient to capture the changes in workload characteristics and to guarantee thermal constraints.

We solve the optimization problem using the Matlab CVX [1]. The policy takes an average of 1 second on a computer with Intel i3 3.3GHz processor when solving for a data center of 360 servers. A 1-second overhead on an average desktop demonstrates that the algorithm can run sufficiently often without noticeable overhead in a data center environment.

VI. Experimental Results

During our evaluation, we use data center simulations based on the linear data center model (Section IV-C), and power, BIPS and temperature data from real-life experiments.

Policy	Normalized BIPS	$\max(T_{cpu})$	$\max(T_{inlet})$	T_{crac}	Efficiency
Server Inlet (SI)	1	59.7°C	24.0°C	17.9°C	73%
CoolBudget (CB)	1.21	70.2°C	33.0°C	26.8°C	84%
ideal CB	1.28	75.0°C	37.9°C	31.7°C	88%

TABLE I. COMPARISON OF SI AND CB, AVERAGED OVER 100 RANDOM SIMULATION SNAPSHOTS

We assume that job arrival times in our data center follow a Poisson distribution with a mean rate of 1 job per second and a mean service time of 3 minutes, resulting in an approximate typical utilization of 50%. An incoming job is randomly selected from our database, and as a thermally-aware job allocation policy, it is allocated to the idle server whose temperature is the least affected by other servers. This corresponds to the idle server with the least row sum in **D**.

We select the highest allowed processor temperature $T_{redline}$ as 75°C, based on two reasons: (1) Experimental analysis on our server shows that leakage power surpasses the fan power when $T_{cpu} > 75$°C, and therefore may adversely affect the energy savings; (2) it is desirable to operate with a margin from reliability-critical temperatures (e.g., 85-90C) of the CPUs to avoid throttling or accelerated failure probabilities.

To evaluate our policy, first, we show the energy savings obtained only by collapsing the thermal headroom margin between server inlets and server internals (in our case, CPUs). Second, we compare CoolBudget (CB) with a state-of-the-art policy to demonstrate the savings achieved by optimal budgeting, and show the impact of thermally-aware job allocation.

Collapsing the thermal headroom in servers: We compare our approach to a policy called server inlet based power budgeting (SI), where the only difference is that SI limits the server inlet temperature, T_{inlet}, instead of limiting T_{cpu}. As the CPU redline 75°C is the worst-case CPU temperature in our server at 24°C inlet and 2400 rpm fan speed, we select the T_{inlet} redline as 24°C.

With our default settings, SI cannot always find a feasible solution for the power constraints where CB displays no performance degradation. In other words, CB enables the support of much lower power limits. To be able to compare the two policies, we use a data center with reduced heat recirculation, where the recirculation matrix **D** is magnitude-wise halved.

Table I shows the simulation results with reduced recirculation and with $P_{budget} = 230kW$, where *ideal CB* assumes perfect (zero error) modeling of the temperature and performance, and the BIPS results are normalized with respect to SI. Due to the over-prediction in the T_{cpu} estimation, CB leaves a temperature headroom of 4.8°C on the average. Increasing T_{crac} by 8.9°C leads to 21% increase in BIPS and 15% improvement in the efficiency ($P_{compute}/P_{budget}$).

Optimal budgeting: We compare our policy with a prior approach called self-consistent budgeting (SC) proposed by Zhan et al. [15]. SC allocates a sufficient amount of power for cooling and budgets the remaining power among the servers. It does not, however, cool down the hottest servers to reduce the cooling need as our policy does. To focus on this difference, we modify SC to use continuous power capping, CPU temperature as the redline, and fair speedup as the objective function.

Figure 4 shows the comparison of SC and CB in both thermally-efficient and random job allocation scenarios. By

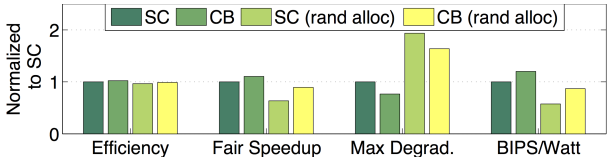

Fig. 4. Comparison between self-consistent (SC) and CoolBudget (CB) policies under two different job allocation scenarios with $P_{budget} = 200kW$

redirecting more of the available power to computing, CB improves the fair speed-up by 10% and BIPS per Watt by 20% during efficient allocation. Random allocation does not affect the efficiency significantly; however, it decreases the fair speedup by 19-36% and the maximum degradation by more than 50%. This is because both SC and CB allocate lower power to thermally inefficient servers to keep a high efficiency, degrading the performance of the jobs in these servers.

VII. CONCLUSION

Effective power budgeting techniques are required to reduce the data center electricity costs and to improve power efficiency. In this paper, we have presented a novel data center power budgeting policy that optimally partitions the given power limit across the servers and the CRAC units. Our policy uses temperature and power models to safely collapse the thermal headroom margins in the servers, and maximizes the overall throughput without an unfair performance degradation across the jobs. Experimental evaluation based on real servers shows that our policy achieves 21% higher power throughput compared to the conventional power budgeting techniques.

REFERENCES

[1] *CVX Research, Inc.* http://cvxr.com/cvx/.

[2] *TileFlow*. http://inres.com/products/tileflow.

[3] C. Bienia. *Benchmarking Modern Multiprocessors*. PhD thesis, Princeton, NJ, USA, 2011. AAI3445564.

[4] H. Chen, C. Hankendi, M. C. Caramanis, and A. K. Coskun. Dynamic server power capping for enabling data center participation in power markets. In *ICCAD'13*, pages 122–129, 2013.

[5] R. Cochran, C. Hankendi, A. K. Coskun, and S. Reda. Pack & cap: Adaptive dvfs and thread packing under power caps. In *MICRO-44'11*, pages 175–185, 2011.

[6] X. Fan, W.-D. Weber, and L. A. Barroso. Power provisioning for a warehouse-sized computer. In *ISCA'07*, pages 13–23, 2007.

[7] S. Govindan, J. Choi, B. Urgaonkar, A. Sivasubramaniam, and A. Baldini. Statistical profiling-based techniques for effective power provisioning in data centers. In *EuroSys'09*, pages 317–330, 2009.

[8] K. Gross, K. Whisnant, and A. Urmanov. Electronic prognostics through continuous system telemetry. In *MFPT'06*, pages 53–62, April 2006.

[9] U. Hoelzle and L. A. Barroso. *The Datacenter As a Computer: An Introduction to the Design of Warehouse-Scale Machines*. Morgan and Claypool Publishers, 1st edition, 2009.

[10] J. Moore, J. Chase, P. Ranganathan, and R. Sharma. Making scheduling "cool": Temperature-aware workload placement in data centers. In *USENIX ATC'05*, pages 5–5, 2005.

[11] N. Rasmussen. Calculating total cooling requirements for datacenters. Technical report, Amer. Power Convers., 2007. white paper 25.

[12] SPEC. *The SPEC CPU2006 benchmark*, 2006. Available: http://www.spec.org/cpu2006/.

[13] SPEC. *The SPECpower_ssj2008 benchmark*, 2008. Available: http://www.spec.org/power_ssj2008/.

[14] Q. Tang, T. Mukherjee, S. K. S. Gupta, and P. Cayton. Sensor-based fast thermal evaluation model for energy efficient high-performance datacenters. In *ICISIP'06*, pages 203–208, Oct 2006.

[15] X. Zhan and S. Reda. Techniques for energy-efficient power budgeting in data centers. In *DAC'13*, pages 176:1–176:7, 2013.