Energy-Proportional Computing: A New Definition

Rathijit Sen David A. Wood
Department of Computer Sciences
University of Wisconsin-Madison

***Abstract*—The original definition of energy-proportional computing, first introduced by Barroso and Hölzle, does not characterize the energy efficiency of recent reconfigurable computers, resulting in non-intuitive "super-proportional" behavior. This paper proposes a new definition of "ideal" energy-proportional computing and new metrics to help design and configure systems to operate close to this ideal efficiency.**

> *"We see that peak energy efficiency occurs at peak utilization and drops quickly as utilization decreases."*
> –Luiz André Barroso and Urs Hölzle[1]

Maximizing energy efficiency $\left(\text{that is, } \frac{\text{Work}}{\text{Energy}}\right.$ or equivalently, $\left.\frac{\text{Performance}}{\text{Power}}\right)$ has economic and environmental benefits as it minimizes the energy needed to do a given computation. Barroso and Hölzle observed that real systems—at that time—attain peak efficiency at peak utilization, but quickly lose efficiency as utilization drops as they are unable to proportionately reduce power consumption. They posit that an "ideal" energy-proportional system should always use energy in proportion to the work done, by maintaining this peak efficiency even at reduced load.

Figure 1 illustrates this original model for a rack-mounted single-node Supermicro 5018D-MTF server with one 4-core Intel Xeon E3-1275 v3 processor[2] (launched: Q2'13, current status: "End of Life") that is running the load-varying SPECpower[3] workload. Figure 1(upper) shows the server's power-performance profile at different load levels with the highest processor frequency. We label these points with *Peak Performance Configuration* since the machine can serve maximum load (peak performance) with this configuration. The *EP* line represents Barroso and Hölzle's ideal energy-proportional profile,[1] where performance is linearly proportional to power. We consider this a *design ideal for* future systems, since current systems have unavoidable idle power consumption. The *Dynamic EP* line accounts for idle power,[4] and represents an *operational ideal* for the current system. This server's Peak Performance Configuration achieves power-performance very close to Dynamic EP. Figure 1(lower) shows that the corresponding energy efficiency (η), normalized to that at peak performance, reduces quickly from 100% as performance drops. In contrast, an EP system is always 100% efficient.

Barroso and Hölzle's observation has been instrumental in helping drive recent system designs to have lower idle power and a wide dynamic power range. However, their model describes systems with *fixed resources*, while these

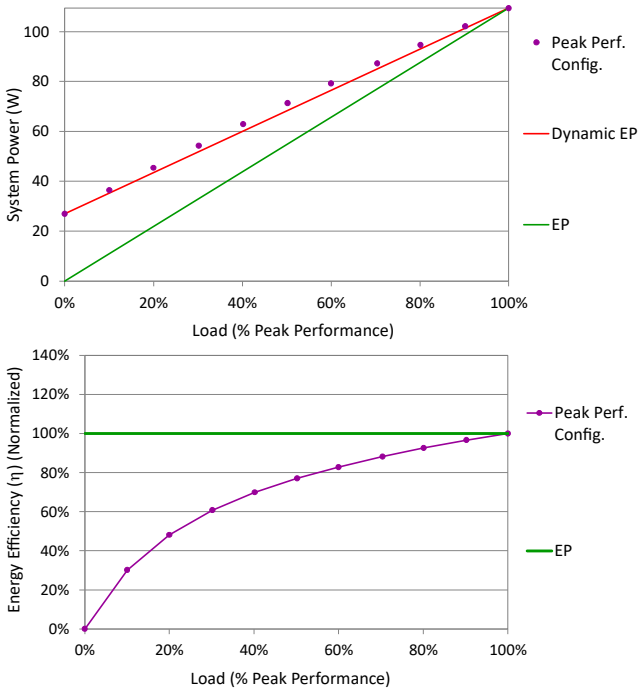

Figure 1: The EP and Dynamic EP models for energy efficiency best describe ideal behavior for systems with fixed resources.

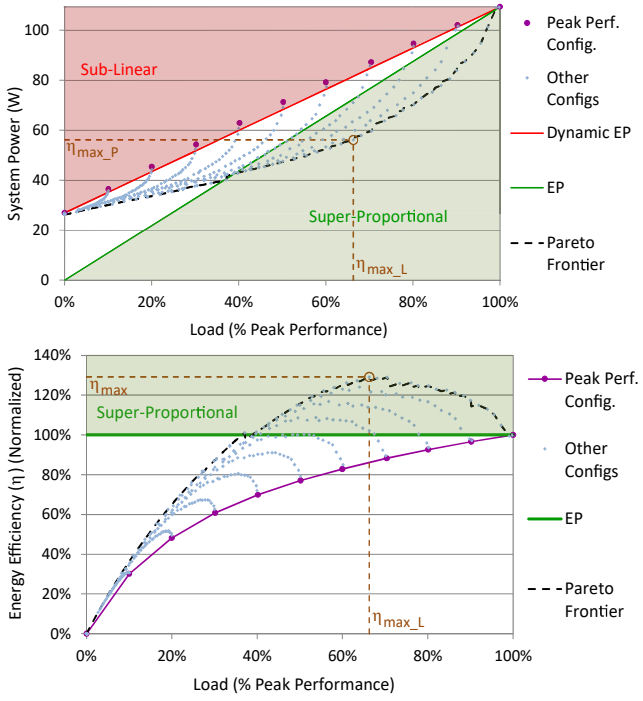

Figure 2: Power-performance and energy efficiency profiles for super-proportional systems with reconfigurable resources. EP and Dynamic EP do not describe ideal behavior for such systems.

SPECpower Overview

This Java workload simulates warehouse transaction processing, with (by default) as many warehouses as logical processors on the system under test, i.e., the server. Transaction requests to each warehouse arrive in batches with (negative) exponentially distributed interarrival times. The server load is measured in total transactions per second. The workload first calibrates the maximum, or 100%, load. Next, it does measurement intervals at load levels 100%, 90%, . . ., 0%. In these intervals, the load served must be within 2% (up to 2.5% shortfall for the 100% and 90% intervals is allowed) of the offered load. The workload measures full system power (at the wall).

We refer to 100% load as the maximum load achieved for the Peak Performance Configuration (all cores at the highest frequency and prefetching enabled). All loads are normalized with respect to that peak load.

modern, more-efficient processors have *reconfigurable resources*—e.g., core frequencies, voltages, number of active cores, threads per core, etc.—that can be varied at runtime.

Operating with fixed resources can be inefficient when a server faces variable loads, either due to fluctuating demands, or service consolidation and load balancing[5, 6, 7] among other servers. Servers are usually configured for maximum performance (that is, the Peak Performance Configuration), but other configurations can trade performance for greater energy efficiency. Figure 2(upper) shows that changing the socket frequency (and consequently voltage) results in energy efficiency that exceeds the EP profile. Specifically, by varying the frequency from 3.9 to 0.8 GHz, this server can achieve higher efficiency over almost 60% of the performance range (points in the shaded Super-Proportional region—where performance is super-proportional to power). Figure 2(lower) shows that the maximum efficiency (η_{max}, occurring at approximately two-thirds load) is 29% higher relative to the EP energy efficiency, for this server.

Reconfigurable systems create opportunities for increased efficiency even outside the super-proportional region. For example, Figure 2(lower) shows that the Peak Performance Configuration has a relative efficiency of 61% at 30% load, but a different configuration attains a relative efficiency of 88% for that load. Thus, the usual server configuration uses 44% more energy than necessary to satisfy the load, despite being nearly on the Dynamic EP line.

Clearly, neither EP nor Dynamic EP describes the full potential of modern computing systems. While non-linearity with reconfiguration is well-known, e.g., with frequency (and voltage) control, the existing ideal models do not consider its impact on peak efficiency.

Redefining EP and Dynamic EP

The EP model assumes that maximum energy efficiency occurs at maximum (100%) load and argues that an ideal system should achieve that efficiency for all loads. Yet Figure 2 shows that a reconfigurable server actually attains maximum efficiency (η_{max}) at a lower load ($\eta_{max\_L} < 100\%$). We argue that a better ideal model is one that achieves this optimal efficiency η_{max} for all loads.

We call this new model *EOP* (Energy Optimal Proportional) since it is both optimal and proportional. EOP is a *design ideal* that gives system designers a way to measure how far the energy efficiency of a target design differs from the best possible design, hopefully leading to more energy-efficient systems. EOP subsumes the EP model for all systems—it improves upon EP for super-proportional systems and is identical to it for all others.

Of course real systems are unlikely to achieve this design ideal, e.g., due to unavoidable idle power, so system software needs an operational model that characterizes the maximum efficiency that can be realized by the current system at different loads. We address this using the well-known power-performance *Pareto frontier*,[8, 9] shown as a dashed line in Figures 2–4. The Pareto frontier represents configurations in the current system that use the lowest power, and hence are the most efficient, among all configurations that can serve a given load.

We call this model *Dynamic EO*. Like Dynamic EP, it is an operational ideal that seeks to characterize the best energy efficiency that can be achieved for a given system. But it differs from Dynamic EP in two aspects—it characterizes optimality that can already be realized by some among the multitude of configurations in the current system and it does not assume linearity of the power-performance profile.

Figure 3 illustrates the different models. These are the

- *design ideals*: conventional (EP), new (EOP), and
- *operational ideals*: conventional (Dynamic EP), new (Dynamic EO).

The EOP line meets (is tangential to) the Dynamic EO line only at points having the maximum efficiency (η_{max}). The following energy efficiency relations hold for any system: Dynamic EP ≤ EP ≤ EOP and Dynamic EO ≤ EOP where ≤ means less than or equal to for values of efficiency. Systems, like our server, that can operate in the non-Sub-Linear region for any portion of their performance range have Dynamic EP ≤ Dynamic EO for all such loads.

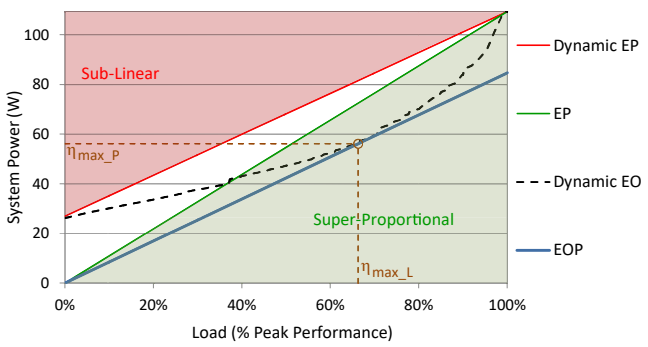

Figure 3: Existing (EP, Dynamic EP) and new (EOP, Dynamic EO) ideal models for energy efficiency.

Power-Performance Pareto Frontier (Dynamic EO)

Every system configuration, or state, can be characterized by its performance and power consumption. The Pareto frontier is the subset of system states that are Pareto-optimal, that is, for which no other state exists with higher performance for at most that power or lower power for at least that performance.

Metric-optimizing states, for example, the most energy-efficient state (efficiency η_{max}), the highest performing state with/without a maximum power cap, the lowest power state with/without a minimum performance bound, the highest performance-per-watt state, the lowest energy state, the lowest energy-delay state, etc. will always lie on the Pareto frontier.

Pareto-optimal states have the same total order in both power and performance. So, increasing the power budget will improve performance at the Pareto frontier if the power is used. This is not true for the whole state space where states with less performance can use more power.

Computational PUE

A hypothetical ideal system, i.e., one that meets the design ideal EOP, achieves maximal energy efficiency (ηmax) and thus minimizes the energy (E_{min}) needed for a given computation regardless of load. We would like a metric to quantify the excess energy used by a real system, compared to this ideal system.

Our new metric, *Computational Power Usage Effectiveness* (or, CPUE), measures how much energy a server uses with configuration c at load l compared to the energy used by EOP. We define

$$\text{CPUE}(c, l) = \frac{\text{Actual server energy with } c \text{ at } l}{\text{EOP energy at } l}, \quad l > 0$$

$$= \frac{E(c, l)}{E_{min}}, \quad l > 0$$

$$\text{Thus, } \boxed{E(c, l) = \text{CPUE}(c, l) \times E_{min}}, \quad l > 0$$

CPUE(c, l) is inspired by the well-known PUE metric[10] that tracks energy waste for datacenters by taking the ratio of facility energy consumption to energy consumption by IT equipment. PUE > 1 quantifies excess relative energy used by the datacenter due to the non-IT infrastructure. Similarly, CPUE(c, l) > 1 quantifies excess computational energy used, relative to E_{min}, whenever efficiency drops below η_{max}.

We have seen that there are two major factors that lead to energy inefficiencies: i) running the system at a non-optimal load and ii) for a given load, running the system with a non-optimal configuration. We can decompose CPUE(c, l) to isolate these two factors.

We defined CPUE(c, l) as $E(c, l)/E_{min}$. For a given amount of work, energy consumed is inversely proportional to efficiency. Thus,

$$\text{CPUE}(c, l) = \frac{\eta_{max}}{\eta(c, l)}, \quad l > 0$$

$$= \left(\frac{\eta_{max}}{\eta_{Pareto}(l)}\right) \times \left(\frac{\eta_{Pareto}(l)}{\eta(c, l)}\right), \quad l > 0$$

$$= \text{LUE}(l) \times \text{RUE}(c, l), \quad l > 0$$

$$\text{Thus, } \boxed{E(c, l) = \text{LUE}(l) \times \text{RUE}(c, l) \times E_{min}}, \quad l > 0$$

where LUE(l) denotes Load *Usage Effectiveness* at load l and RUE(c, l) denotes *Resource Usage Effectiveness* of configuration c at load l.

LUE(l) is the efficiency of EOP (η_{max}) relative to that of Dynamic EO at load l. LUE(l) ≥ 1 with LUE(l) = 1 $\Longleftrightarrow l$ can be served at maximum efficiency (η_{max}). Since energy consumed is inversely proportional to efficiency, LUE(l) > 1 quantifies excess energy used, relative to E_{min}, due to non-optimal loads assuming that the Pareto-optimal configuration has been chosen to serve load l.

RUE(c, l) is the efficiency of Dynamic EO relative to that of configuration c, both at load l. RUE(c, l) ≥ 1 with RUE(c, l) = 1 $\Longleftrightarrow$ c is a Pareto-optimal configuration. RUE(c, l) > 1 quantifies excess energy used, relative to Dynamic EO, due to using non-optimal (Pareto-dominated) configuration c for serving load l.

Both LUE(l) and RUE(c, l) can be expressed in terms of CPUE(c, l). Since $\text{RUE}_{Pareto}(l) = 1$ for every l, LUE(l) = $\text{CPUE}_{Pareto}(l)$ and RUE(c, l) = CPUE(c, l)/$\text{CPUE}_{Pareto}(l)$.

Calculating LUE and RUE (as well as determining EOP and Dynamic EO) requires knowledge of the Pareto frontier. In this work, we determine the frontier offline by running the workload multiple times with the server configured to different frequencies. Offline characterization is also used in prior work,[8, 9] but may not be feasible in an online setting with unknown workloads. In the next sections we introduce an online policy that closely approximates the frontier by controlling processor frequency and cache prefetching.

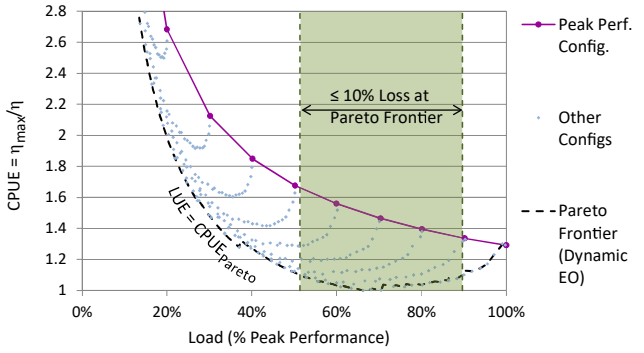

Figure 4: New energy efficiency metrics, CPUE(c, l) and LUE(l) for configuration c and load l. CPUE >1 indicates suboptimal c and/or l whereas LUE >1 indicates suboptimal l but Pareto-optimal c for that l. These metrics → ∞ as l → 0 due to non-zero idle power. CPUE(c, l) ≥ $\text{CPUE}_{Pareto}(l)$ = LUE(l).

Related Work in Energy Efficiency Characterization

Barroso and Hölzle[1] were the first to propose the idea of energy proportionality. They compute power efficiency as $\frac{\text{Utilization}}{\text{Power}}$. We use the same definition for efficiency, but redefine utilization as percentage of peak performance.

David Lo et al.[2] proposed a relaxed model of energy proportionality, called Dynamic Energy Proportionality, that ignores idle power. This corresponds to the Dynamic EP line in Figure 1. This linear model has also been studied in other prior works.[3, 4] Daniel Wong and Murali Annavaram[4] name the region that we call Sub-Linear as Superlinear. We prefer to use "Sub-" in the sense that operating in this region lowers efficiency compared to that on Linear (Dynamic EP).

Chung-Hsing Hsu and Stephen W. Poole[5] observed real machines doing better than the conventional design ideal, EP, that assumes linear proportionality. They proposed quadratic proportionality as the new ideal. However, this makes ideal system efficiency load-dependent, with higher efficiency at lower loads than at higher loads.

Our view is that the design ideal, EOP, will have maximum efficiency (η_{max}) independent of load and will consume power linearly proportional to load, as proposed in the original EP model, but the constant of proportionality is different: it is defined by the most efficient configuration instead of by the peak performance configuration. The Pareto frontier (Dynamic EO) is the operational ideal for the system and its efficiency is load-dependent. The most efficient configurations lie at the intersection of the EOP and Dynamic EO curves.

A number of metrics for characterizing energy efficiency exist. Efficiency $\left(\frac{\text{Performance}}{\text{Power}}\right)$ can be computed at individual loads,[1] or as $\frac{\text{Total Performance}}{\text{Total Power}}$ over all loads.[6] Metrics based on the dynamic power range compute the ratio between the idle and peak power consumptions.[3] Other metrics consider the deviation of the power curve from an ideal curve, e.g., maximum relative power difference with respect to Dynamic EP,[3] area enclosed by the power curve relative to that by Dynamic EP[4] or EP,[4, 7] power used in excess to that by EP,[4] etc. These metrics continue to be useful with the new ideals, EOP and Dynamic EO, replacing the conventional ideals.

Song et al.[8] proposed Iso-energy-efficiency (EE) as the energy ratio between sequential and parallel executions of a given application. EE aims for equal efficiency as a system scales up whereas EP and EOP aim for equal efficiency as a system experiences variable loads. Our CPUE, LUE, and RUE metrics use EOP or Dynamic EO for reference instead of specific execution modes and consider the load along with the configuration for quantifying losses.

Barroso and Hölzle[9] compute datacenter energy consumption as PUE × SPUE × energy to electronic components. While PUE[10] accounts for non-IT overheads in datacenter infrastructure, SPUE (Server PUE) accounts for overheads within a server, e.g., power supply unit (PSU) losses. CPUE, LUE, and RUE do not separate SPUE losses from computing energy but separate energy-wasting configurations and loads from optimal ones.

References

1. L. A. Barroso and U. Hölzle, "The Case for Energy-Proportional Computing," *Computer*, vol. 40, no. 12, pp. 33–37, Dec 2007.

2. D. Lo, L. Cheng, R. Govindaraju, L. A. Barroso, and C. Kozyrakis, "Towards Energy Proportionality for Large-scale Latency-critical Workloads," in *ISCA*. IEEE Press, 2014, pp. 301–312.

3. G. Varsamopoulos and S. K. S. Gupta, "Energy Proportionality and the Future: Metrics and Directions," in *ICPPW*. IEEE Computer Society, 2010, pp. 461–467.

4. D. Wong and M. Annavaram, "KnightShift: Scaling the Energy Proportionality Wall Through Server-Level Heterogeneity," in *MICRO*. IEEE Computer Society, 2012, pp. 119–130.

5. C.-H. Hsu and S. Poole, "Revisiting Server Energy Proportionality," in *ICPP*. IEEE Computer Society, 2013, pp. 834–840.

6. "Specpower_ssj," 2008, https://www.spec.org/ power_ssj2008/

7. F. Ryckbosch, S. Polfliet, and L. Eeckhout, "Trends in Server Energy Proportionality," *Computer*, vol. 44, no. 9, pp. 69–72, Sept 2011.

8. S. Song, C.-Y. Su, R. Ge, A. Vishnu, and K. W. Cameron, "Iso-Energy-Efficiency: An Approach to Power-Constrained Parallel Computation," in *IPDPS*. IEEE Computer Society, 2011, pp. 128–139.

9. L. A. Barroso and U. Hölzle, *The Datacenter as a Computer: An Introduction to the Design of Warehouse-Scale Machines*, 1st ed. Morgan and Claypool Publishers, 2009.

10. V. Avelar, D. Azevedo, and A. French, Eds., *PUE: A Comprehensive Examination of the Metric*. The Green Grid, 2012.

Load Management: reduce LUE

Most data centers are provisioned to meet peak load, but normally operate at much lower load levels. The LUE metric can help operators quantify the potential benefit of deploying load management policies,[5, 6, 7] e.g., concentrating load on some servers and shutting down others. Of course, any such policy must also ensure that service-level agreements are still satisfied.

Figure 4 shows that CPUE for the Peak Performance Configuration is always > 1 (wastes energy) and increases as load decreases. The best CPUE for this configuration is 1.29, occurs at peak load, and implies 29% excess energy used relative to E_{min}. LUE (that is, CPUE for Dynamic EO), on the other hand, first decreases to 1, then increases, revealing a sweet spot of ≤ 10% excess energy used at around 51%–90% of peak performance.

Barroso and Hölzle[1] observed that servers typically operate at 10%–50% load. The LUE curve for SPECpower (Figure 4), shows excess energy used due to suboptimal load of approximately 10% at the higher end of this range, to over 250% (not shown) at the lower end. The steep slope

of the LUE curve at low loads makes even modest load management very attractive. For example, increasing load from 10% to 20% of peak reduces LUE from 3.55 (255% excess) to 1.99 (99% excess) and a further increase to 25% peak load reduces LUE to 1.68 (68% excess).

Configuration Management: reduce RUE

Even in a data center with perfect load balancing, reconfigurable servers may be misconfigured, wasting significant energy even at optimal load. Figure 5 shows RUE for SPECpower for all system configurations (DVFS) and loads. Operating with the Peak Performance Configuration is significantly wasteful, e.g., 21% excess energy used at 10% load compared to operating at Dynamic EO. The excess increases to 51% before decreasing to zero at peak load. Not all Pareto-dominated configurations are as wasteful—the shaded band identifies configurations that have an RUE of ≤ 1.1 and hence limit the extra energy used to 10%.

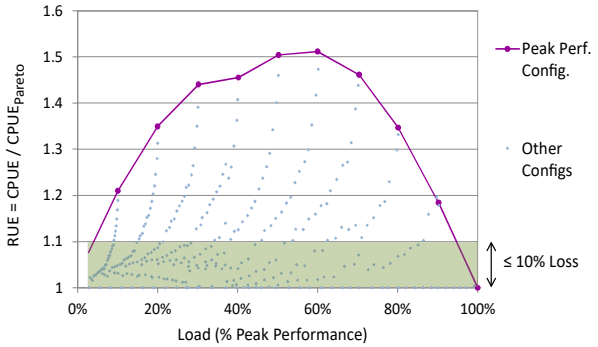

Figure 5: New energy efficiency metric, RUE(*c, l*), for configuration *c* and load *l*. RUE >1 indicates suboptimal *c* for that *l*. RUE(*c, l*) = CPUE(*c, l*)/CPUE$_{Pareto}$(*l*) = η$_{Pareto}$(*l*)/η(*c, l*).

Linux includes various governors[11] that govern processor frequency by either setting it to a fixed value or varying it in response to processor utilization. In Figure 6, we use the prefix 'L-' to label the power-performance profiles of SPECpower with Linux governors *performance*, *ondemand*, *conservative*, and *powersave*. L-powersave lies fully on Dynamic EO, but can serve only up to 27% of peak load. The other governor profiles are distant from Dynamic EO, thus having large RUE values.

Spiliopoulos, Kaxiras, and Keramidas proposed green governors[12] that predict performance and power for various frequencies and select the best configuration. A simpler reactive governor, R(*t,p*), works well for SPECpower. Figure 6 shows that R(100,20) has low RUE throughout the performance range and even exceeds the former peak performance, by around 4%, due to cache prefetch control.

This work explores the relation between two well-known but dissimilar concepts, (power-performance) Pareto optimality and energy proportionality, both of which share the end goal of making computing more energy efficient. While system components are increasingly being designed to be reconfigurable, identifying the Pareto frontier is challenging, particularly with multiple reconfigurable resources and dynamically changing runtime environments. Scheduling frameworks that carefully choose configurations and operating ranges will unlock the full potential of current and future reconfigurable systems.

Reactive, R(*t,p*), governor design

Within repeating intervals of *t* milliseconds each, it

1. profiles performance with L2 cache prefetching turned off and on for *p/2* milliseconds each and chooses the better-performing prefetch mode, and
2. selects the highest frequency (0.5 GHz granularity) that it predicts will not result in idle cycles by dividing the rate per second of active cycles (CPU_CLK_UNHALTED.THREAD) with the number of logical cores. To react to sudden performance demands, it increases frequency in doubling steps (0.1 GHz starting step size) over consecutive intervals if the target frequency is not less than the current frequency.

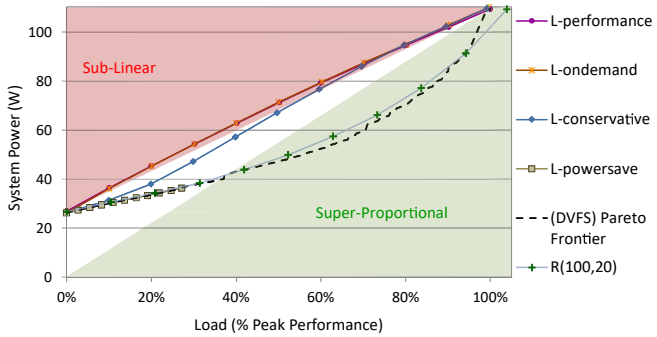

Figure 6: Existing and new governors. Our new reactive governor, R(*t,p*), causes the system to operate closest to the Pareto Frontier.

Acknowledgments

We thank Luiz Barroso, Edouard Bugnion, Mark Hill, Michael Marty, Kathryn McKinley, Michael Swift, the Editor, and anonymous reviewers for helpful comments. This work was supported in part with NSF grants CCF-1218323 and CNS-1302260. The views expressed herein are not necessarily those of the NSF. Wood has significant financial interests in AMD and Google.

References

1. L. A. Barroso and U. Hölzle, "The Case for Energy-Proportional Computing," *Computer*, vol. 40, no. 12, pp. 33–37, Dec 2007.

2. "Intel Xeon Processor E3-1275 v3", http://ark.intel.com/products/75464/Intel-Xeon-Processor-E3-1275-v3-8M-Cache-3_50-GHz

3. "Specpower_ssj," 2008, https://www.spec.org/ power_ssj2008/

4. D. Lo, L. Cheng, R. Govindaraju, L. A. Barroso, and C. Kozyrakis, "Towards Energy Proportionality for Large-scale Latency-critical Workloads," in *ISCA*. IEEE Press, 2014, pp. 301–312.

5. J. S. Chase, D. C. Anderson, P. N. Thakar, A. M. Vahdat, and R. P. Doyle, "Managing energy and server resources in hosting centers," in *SOSP*. ACM, 2001, pp. 103–116.

6. J. Mars, L. Tang, R. Hundt, K. Skadron, and M. L. Soffa, "Bubble-Up: Increasing Utilization in Modern Warehouse Scale Computers via Sensible Co-locations," in *MICRO*. ACM, 2011, pp. 248–259.

7. C. Delimitrou and C. Kozyrakis, “Quasar: Resource-efficient and QoS- aware Cluster Management,” in *ASPLOS*. ACM, 2014, pp. 127–144.

8. M. Ruggiero, A. Acquaviva, D. Bertozzi, and L. Benini, “Application-specific power-aware workload allocation for voltage scalable mpsoc platforms,” in *ICCD*. IEEE Computer Society, 2005, pp. 87–93.

9. P. E. Bailey, A. Marathe, D. K. Lowenthal, B. Rountree, and M. Schulz, “Finding the limits of power-constrained application performance,” in *SC*. ACM, 2015, pp. 79:1–79:12.

10. V. Avelar, D. Azevedo, and A. French, Eds., *PUE: A Comprehensive Examination of the Metric*. The Green Grid, 2012.

11. D. Brodowski and N. Golde, “CPU frequency and voltage scaling code in the Linux(TM) kernel. Linux CPUFreq. CPUFreq Governors”, https://www.kernel.org/doc/Documentation/cpu-freq/governors.txt.

12. V. Spiliopoulos, S. Kaxiras, and G. Keramidas, “Green governors: A framework for Continuously Adaptive DVFS,” in *IGCC*. IEEE Computer Society, July 2011, pp. 1–8.

Rathijit Sen *is a scientist at the Gray Systems Lab, Microsoft Corporation. His research interests include performance modeling and power management of computer systems. Sen received a PhD in computer science from the University of Wisconsin-Madison where this research was done. Contact him at rathijit.sen@microsoft.com.*

David A. Wood *is the Amar and Balinder Sohi Professor in Computer Science at the University of Wisconsin-Madison. His research interests include parallel and heterogeneous computer system design, memory system design, and computer simulation. Wood received a PhD in computer science from the University of California, Berkeley. He is an ACM Fellow, an IEEE Fellow, and a member of the IEEE Computer Society. Contact him at david@cs.wisc.edu.*

Submitted: October 2015; Revised (Major): March 2016; Revised (Minor): April 2017; Accepted: May 2017

An extended journal paper on this work is published in ACM TACO:

Rathijit Sen and David A. Wood, "*Pareto Governors for Energy-Optimal Computing*", ACM Transactions on Architecture and Code Optimization. Volume 14, Issue 1, Article 6. March 2017. DOI: https://doi.org/10.1145/3046682