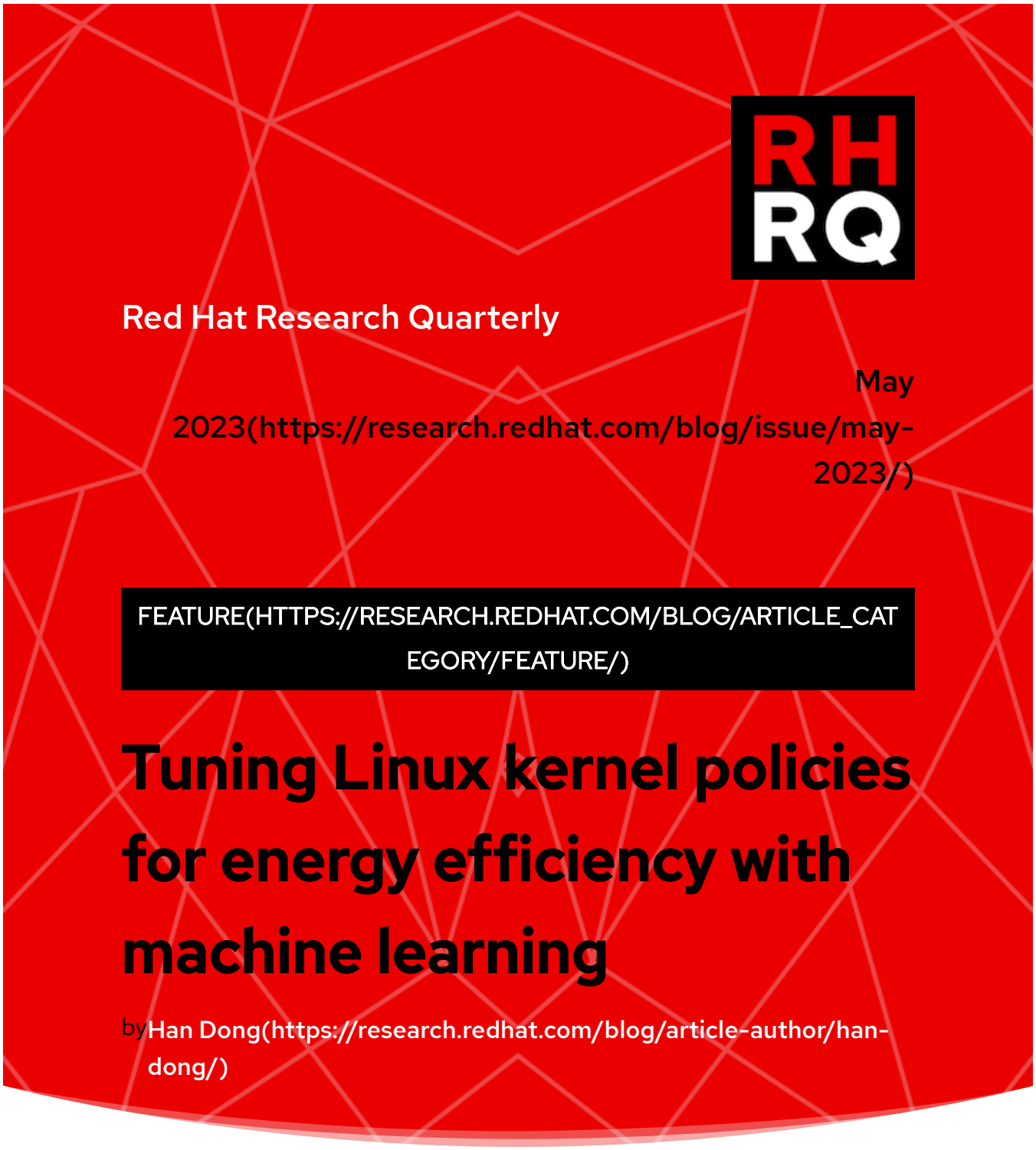


Red Hat Research Quarterly

May 2023(https://research.redhat.com/blog/issue/may-2023/)

FEATURE(HTTPS://RESEARCH.REDHAT.COM/BLOG/ARTICLE_CATEGORY/FEATURE/)

Tuning Linux kernel policies for energy efficiency with machine learning

by Han Dong(https://research.redhat.com/blog/article-author/han-dong/)

Red Hat Research Quarterly

May 2023(https://research.redhat.com/blog/issue/may-2023/)

FEATURE(HTTPS://RESEARCH.REDHAT.COM/BLOG/ARTICLE_CATEGORY/FEATURE/)

Tuning Linux kernel policies for energy efficiency with machine learning

by Han Dong(https://research.redhat.com/blog/article-author/han-dong/)

Search articles

9 min read

Presenting BayOp, a generic ML-enhanced controller that optimizes network application efficiency by automatically controlling performance and energy trade-offs.

As global datacenter energy use rises and energy budgets are constrained, it becomes increasingly important for operating systems (OS) to enable higher efficiency and get more work done while consuming less. Concurrently, the environmental footprint of hardware manufacturing is increasing,[1] further underscoring the importance of extracting more value from existing resources. The complexity of modern systems software and hardware and the end of Dennard scaling and Moore's Law only add to the difficulty. Meeting these challenges will require solving the problem of how to specialize different kernel policies and hardware configurations that are often written to support only general use cases.

Moreover, as latency-critical applications, such as key-value stores (i.e., memcached), become ubiquitous across datacenters, cloud service providers more frequently choose to deploy them on dedicated hardware. Motivated by the rise of this software-hardware dedication, this article presents BayOp. This generic controller optimizes the efficiency of network applications by taming and controlling the system's performance and energy trade-offs automatically. BayOp uses an established machine learning (ML) technique, Bayesian optimization, to exploit two hardware mechanisms that externally control interrupt coalescing and processor energy settings. The key insight behind BayOp's dynamic adaptation is that controlling interrupt coalescing induces batching to stabilize application

latency periods, making it easier to control performance-energy trade-offs and magnifying the benefits of batching with processor energy settings.

Our team of Boston University researchers and Red Hat engineers are making the following contributions toward realizing BayOp:

- We conducted a novel performance and energy study of two network applications by sweeping up to 340 static combinations of (1) a network interface controller's (NIC) interrupt delay setting (ITR) to control interrupt coalescing and (2) the processor's Dynamic Voltage Frequency Scaling (DVFS) to control processor energy settings.
- Our study found that performance improvements of over 74% are possible in Linux for a simple ping-pong network application by using a static ITR. We investigated the performance and energy trade-offs for a memcached server and found that tuning both ITR and DVFS can yield 76% in energy savings.
- Our data also reveal that tuning ITR and DVFS results in stable OS behavior, which implies that this structure can be captured formally. Based on these findings, we developed the BayOp controller, which can dynamically adjust the settings of a memcached server to adapt to changing offered loads and performance and energy goals while meeting different service-level agreement (SLA) objectives.

ITR performance study in NetPIPE

A common feature of modern high-speed NICs is the ability to control batching via interrupt coalescing. In this work, we explore this mechanism on an Intel 82599 10GbE by using its ITR register, which is exposed by Linux ethtool. Software typically uses the ITR register to configure a delay in 2-microsecond (µs) increments from 0 to 1024 µs. If the spacing of interrupts is less than the ITR value, the NIC will delay interrupt assertion until the ITR period has been met. Linux's network device driver typically

contains a dynamic policy that seeks to performance-tune the ITR value to better reflect the current workload.

However, our study reveals that Linux's default dynamic ITR policy can result in performance instabilities even in a simple network ping-pong application such as NetPIPE. Figure 1 illustrates the measured performance, or Goodput, differences for a range of message sizes between Linux, which uses its dynamic ITR policy, and Linux-static, where we disabled its dynamic policy and selected a single fixed ITR value instead. This figure illustrates that using a static ITR was able to achieve higher Goodput in all message sizes. For example, at 64 KB messages, Linux-static improved its performance by 74%.

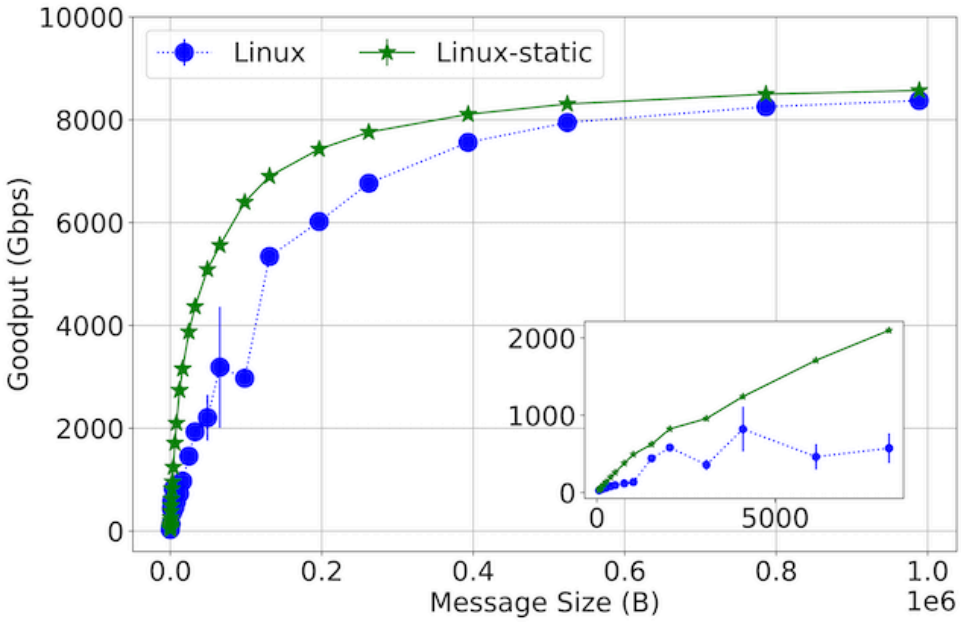


Figure 1. *Goodput measurements for NetPIPE across different message sizes from 64 bytes to 1 MB. The inset is zoomed in on message sizes less than 8 KB. The error bars on each point show the standard deviation of measured performance.*

This performance difference can be traced to the behavior of Linux's dynamic ITR policy. Figure 2 shows a snapshot of every updated ITR value captured in Linux's network device driver during a single run of NetPIPE using 64 KB message sizes. This figure illustrates the extreme variability (up to 500 µs) at which ITR is updated on a per-interrupt basis. This variability suggests that the current dynamic policy, designed to support general use cases, is operating at the wrong timescale for an application such as NetPIPE and that further specialization can yield significant advantages.

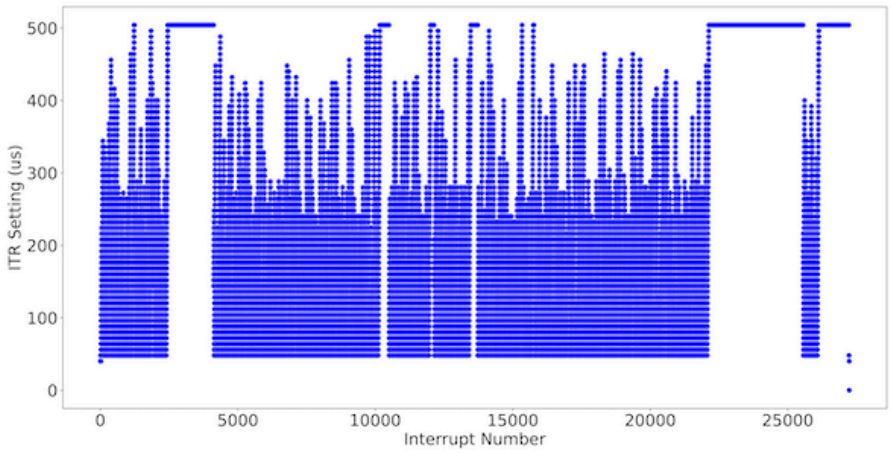


Figure 2. *ITR values set by Linux's dynamic ITR algorithm. This is captured during a live run of NetPIPE at 64 KB message size.*

Performance and energy

We then expanded our study to uncover the performance and energy trade-offs within this space. Toward this end, we explored the novel use of ITR and DVFS by statically configuring up to 340 unique combinations of each across both NetPIPE and a memcached server. In Figure 3 and Figure 4, we compare Linux-static, which statically sets both, against Linux, which

uses the dynamic ITR policy and dynamic DVFS powersave governor. Memcached is an example of an application with an external request rate that can largely be considered independent from the time required to service a request. Service providers often set an SLA target for these types of applications, such as some percentage of requests to be completed under a stringent time budget (99% tail latency < 500 µs for Figure 4). At the same time, there is a constant stream of requests-per-second (QPS) arriving at the server.

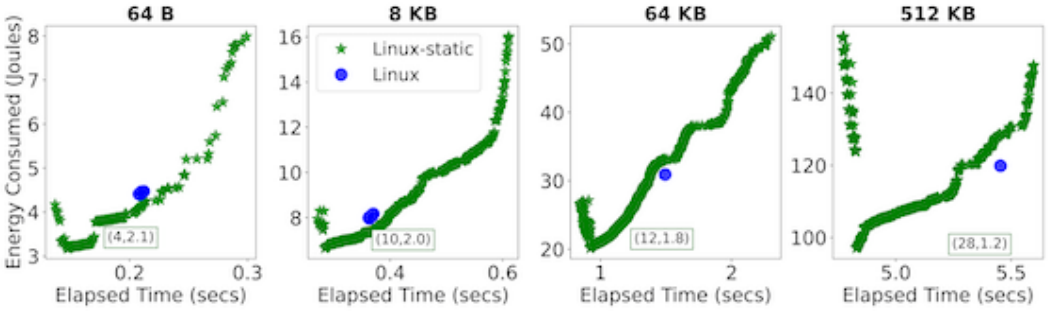


Figure 3. *NetPIPE performance and energy results for different message sizes. Every Linux-static datapoint is the result of a single experimental run with a unique (ITR, DVFS) combination. The X-axis is a measure of performance (lower is better), and Y-axis shows total energy consumed. The labeled (ITR, DVFS) pair are experimental Linux-static values that resulted in lowest energy use. The number of round-trips is fixed at 5000 for each message size.*

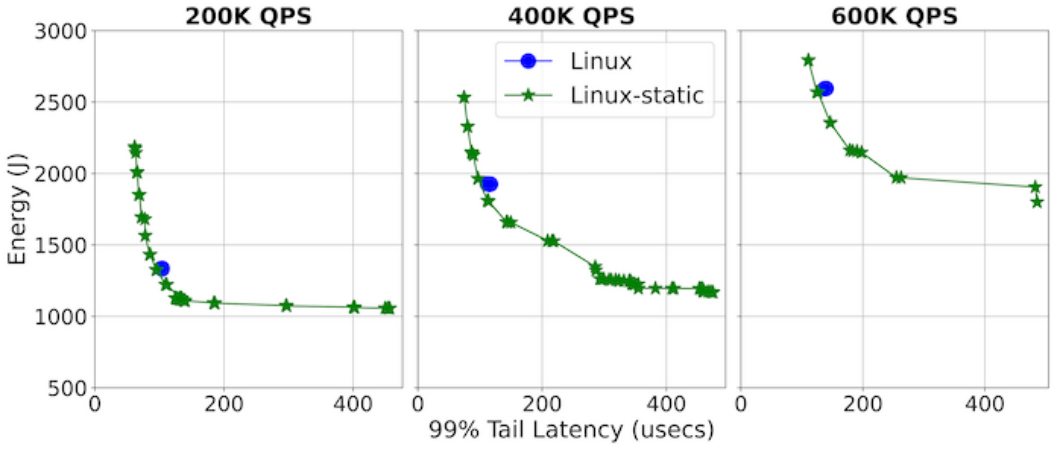


Figure 4. *Each point represents a single experimental run of memcached at different QPSes. Each Linux-static data point uses a unique (ITR, DVFS) pair, and we only illustrate data that lie on the Pareto-optimal curve. The X-axis is a measure of performance (lower is better), and Y-axis shows total energy consumed.*

Figures 3 and 4 illustrate a rich performance and trade-off space between Linux and Linux-static, whereby tuning both ITR and DVFS can result in dramatic energy savings of over 50% and even improve performance in the case of NetPIPE. This study also reveals characteristic shapes of Linux that differ depending on the application. For NetPIPE, there is the V shape. The lowest point in this V shape represents a setting that uses the lowest energy while being competitive in performance; the vertical points above represent other configurations that sacrifice energy for better performance. Note that Linux always lies to the right of the V curve, indicating the value of doing such a static search. In contrast, memcached reveals an L shape. While this L shape differs in absolute performance and energy, the underlying Linux response to changes in (ITR, DVFS) combinations remains stable across the offered loads, which suggests one can capture these behaviors formally.

BayOp design and results

To operationalize these energy savings and stabilize OS response behaviors, we built BayOp, an application- and OS-agnostic controller using a sample-efficient ML technique—Bayesian optimization—to automatically probe for efficient (ITR, DVFS) settings within a running system while under changing offered loads. In particular, we targeted applications such as memcached, where the SLA space enables a rich set of performance and energy trade-offs.

We used BayOp to automatically tune a memcached server while servicing a publicly available memcached trace from Twitter(http://github.com/twitter/cache-trace). Twitter's trace reveals that these services often maintain a mean demand curve that changes slowly over periods of 24 hours or more. These can be attributed to either diurnal access patterns or can be induced through service admission control layers such as load balancers. These curves suggest that specialization of a single application at a specific offered load can be a realistic form of optimization to exploit the stable regions of these demand curves.

Figure 5 illustrates the BayOp controller design. In phase 1, a live system running memcached is currently servicing various QPSes from an external source. BayOp will then periodically trigger a set of performance and energy measurements of the live system in phase 3. For each measurement in phase 3, the Bayesian optimization process is used to compute a reward penalty of its current configuration and then, in phase 4, recommend and update a new (ITR, DVFS) pair on the live system such that it minimizes the reward penalty. Once this process is finished, the memcached server is set with a static (ITR, DVFS) setting until the next set of measurements is triggered.

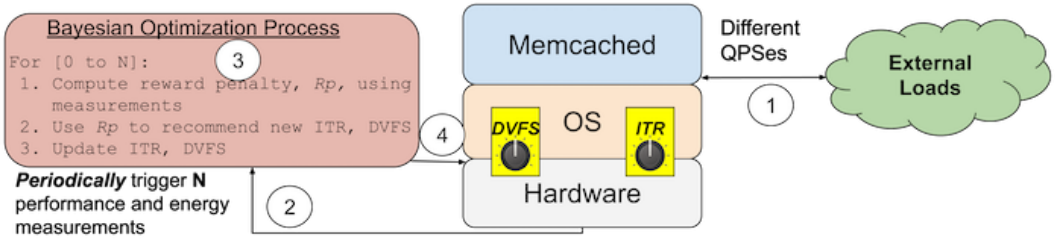
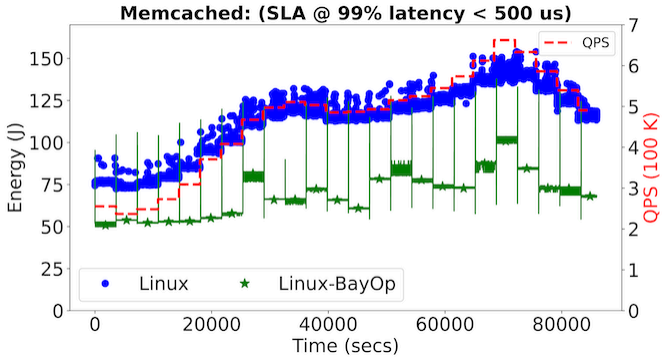
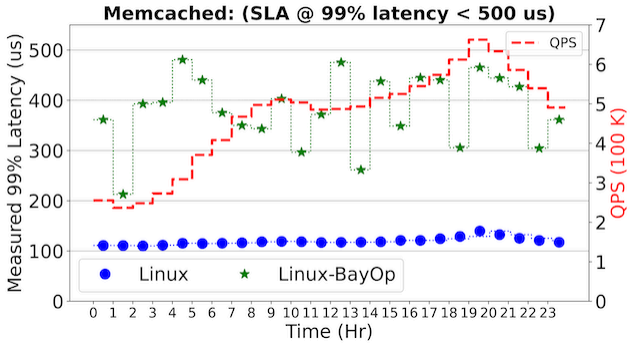


Figure 5. *BayOp controller for a memcached server*

Figure 6 illustrates results from two different SLA objectives: 99% latency < 500 µs, and a stringent 99% latency < 200 µs. The figures to the left illustrate the system's energy usage on a per-second basis over a 24-hour period, and the change in hourly QPS rate is also indicated. At the beginning of each hourly QPS rate, there are spikes in energy usage from Linux-BayOp, the result of the Bayesian optimization process as it dynamically searches through (ITR, DVFS) settings on the memcached server to meet its corresponding SLA objective. After this initial energy spike, the system settles to a steady energy consumption state until the next hourly trigger. The figures to the right illustrate the latency trade-offs BayOp makes to maintain SLA objectives while saving energy.





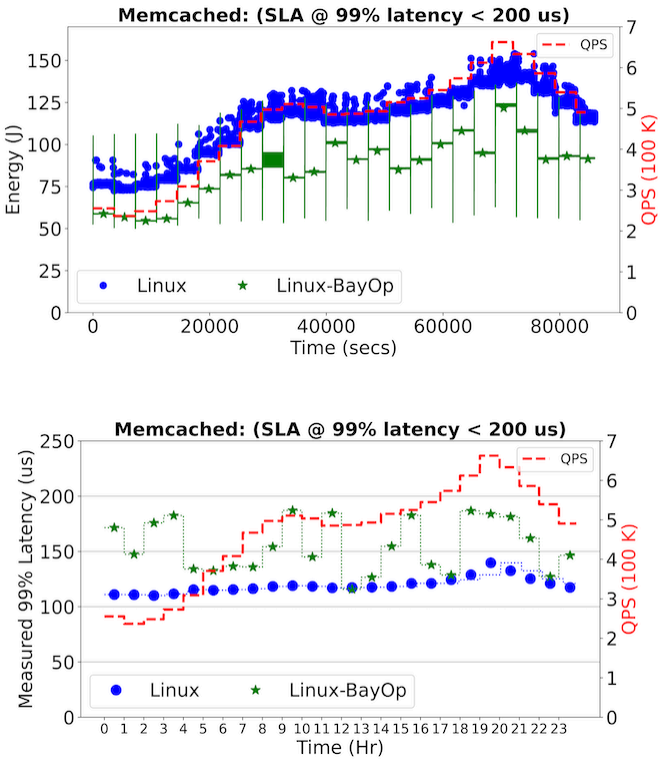


Figure 6. *BayOp applied to cache-trace QPS rates over a 24-hour period for memcached. Each row represents a different SLA objective and shows the measured energy per-second as QPS changes across the five system configurations studied.*

In the case of an SLA objective of 99% latency < 500 µs, we find that Linux-BayOp can result in energy savings of up to 50% over Linux. Even at a more stringent SLA of 99% latency < 200 µs, Figure 6 shows that Bayesian optimization can adapt to this new requirement while saving up to 30% energy. These results demonstrate the generality of the BayOp controller. They also reveal the limitations of Linux's existing dynamic policies, as one cannot use its current ITR and DVFS algorithms to express these rich performance and energy SLA trade-offs.

Future directions for BayOp

BayOp's design as an external controller creates the potential to integrate with load balancers to exploit and optimize a dynamic fleet of servers that direct incoming offered loads to one or more servers configured with specific ITR and DVFS. Additionally, if servers are added or removed from the fleet, BayOp re-optimization can then be coordinated with re-balancing.

In our use of BayOp to optimize memcached servers, we make certain simplifying assumptions, such as the hourly trigger to run the Bayesian optimization process. While we have demonstrated that these simple assumptions can result in significant advantages, there is considerable room for improvement. BayOp's architecture enables the integration of more advanced policies for deciding when to trigger the Bayesian process, such as in response to dramatic changes in QPS rates or exploiting historical patterns in service loads. The Bayesian optimization package can also be improved to reduce the cost of sampling.

Our work is currently being extended through the Red Hat Collaboratory at Boston University to improve performance and energy efficiency for open sourced stream processing applications as well. Reach out to handong@bu.edu(mailto:handong@bu.edu) with questions, see the data collection infrastructure at GitHub(http://github.com/sesa/intlog), or visit the project page(http://Automatic Configuration of Complex Hardware) on the Red Hat Research website.

Footnote

1. "Chasing carbon: the elusive environmental footprint of computing(http://arxiv.org/abs/2011.02839)," IEEE International Symposium on High-Performance Computer Architecture (HPCA 2021).

SHARE THIS ARTICLE

MORE LIKE THIS

FEATURE

Faster hardware through software(https://research.redhat.com/blog/article/faster-hardware-through-software/)

Gordon Haff

Researchers have tested several techniques for using software to get the most out of hardware. Find out about three promising projects that indicate the direction of this quickly changing field. It used to be simple to make computer workloads run faster. Wait eighteen months or so for more transistors consuming the same amount of power, [...]

FEATURE

Isn't multi-tenancy Ironic? (https://research.redhat.com/blog/article/isnt-multi-tenancy-ironic/)

Tzu-Mainn Chen

Lars Kellogg-Stedman

Virtualization is an amazing technology that has become a popular solution for sharing resources among members of an organization. However, some organizations need to harness the capabilities of an entire machine, without a layer of virtualization between the code and the hardware. Is it possible to share hardware between projects with the

same ease as sharing virtual resources?

FEATURE

Testing critical IoT systems to mitigate network disruptions(https://research.redhat.com/blog/article/testing-critical-iot-systems-to-mitigate-network-disruptions/)

Miroslav Bureš

The Internet of Things brings new opportunities and new challenges for mission-critical applications where lives are at stake. Systematic testing can help. The Internet of Things (IoT) has significantly increased the capabilities of mission-critical systems in many domains. Integrated rescue systems, healthcare, defense, energy, and transportation benefit from

FEATURE

RISC-V for FPGAs: benefits and opportunities(https://research.redhat.com/blog/article/risc-v-for-fpgas-benefits-and-opportunities/)

Ahmed Sanaullah

Why open source hardware will play a key role in emerging technologies research RISC-V Instruction Set Architecture (ISA)-based microarchitectures are an important part of all Field Programmable Gate Array (FPGA)-based research projects in the Red Hat Collaboratory at Boston University. Having CPU cores in FPGA designs is important: partitioning workloads

using the IoT, enabling faster system reactions [...]

between special purpose FPGA circuits [...]

FEATURE

Applying lessons from our upstream hypervisor fuzzer to improve kernel fuzzing(https://research.redhat.com/blog/article/applying-lessons-from-our-upstream-hypervisor-fuzzer-to-improve-kernel-fuzzing/)

Alexander Bulekov

Bandan Das

Could a grammarless approach increase its effectiveness? Low-level systems such as Linux kernels and hypervisors form the foundation of cloud systems today. The virtual machines (VMs) provided by hypervisors are attractive targets for attackers. Bugs in hypervisors create the risk of

FEATURE

Demystifying real-time Linux scheduling latency(https://research.redhat.com/blog/article/demystifying-real-time-linux-scheduling-latency/)

Daniel Bristot de Oliveira

This is the third of a series of three articles about the formal analysis and verification of the real-time Linux® kernel. Read the first article in RHRQ 2:3 and the second article in RHRQ 2:4.

an attacker in a malicious VM, compromising the isolation guarantees provided by the hypervisor, [...]

FEATURE

Fostering open innovation in hardware(https://research.redhat.com/blog/article/fostering-open-innovation-in-hardware/)

Yan Fisher

Why is open hardware important? How is the new RISC-V architecture bringing open hardware research to the forefront? How will this impact you? Read on to find out.

FEATURE

Can streaming data and machine learning build better communities? (https://research.redhat.com/blog/article/can-streaming-data-and-machine-learning-build-better-communities/)

Jim Craig

An open source powered smart village project underway at the Red Hat Collaboratory may have the potential to change the world —or at least a town near you. For as long as I can remember

—and after almost 40 years in the IT industry, that's quite a while now—every year for the last 20 years or so [...]

FEATURE

When machine learning meets big data processing: From human-native tasks to machine-native tasks(https://research.redhat.com/blog/article/when-machine-learning-meets-big-data-processing-from-human-native-tasks-to-machine-native-tasks/)

Ilya Kolchinsky

Since the inception of artificial intelligence research,

computer scientists have aimed to devise machines that think and learn like human beings. What else could AI do?

ABOUT THE AUTHOR

Han Dong

Han Dong is a postdoc in the Computer Science department at Boston University. His research interests lie in distributed systems, high-performance computing, and operating systems. He is interested in research addressing the growing energy needs of our modern systems.

RELATED PROJECTS

- Automatic Configuration of Complex Hardware(https://research.redhat.com/blog/research_project/automatic-configuration-of-complex-hardware/)

ARTICLE FEATURED IN

(https://research.redhat.com/blog/issue/may-2023/)

(https://research.redhat.com/blog/issue/may-2023/)

Red Hat Research Quarterly

May 2023

Download PDF

Subscribe now

IN THIS ISSUE

INTERVIEW

“That’s what open source is all about”: A short history of collaboration, innovation, and education in research(https://research.redhat.com/blog/article/thats-what-open-source-is-all-about-a-short-history-of-collaboration-innovation-and-education-in-research/)

Shaun Strohmer

FEATURE

Meet CCO: a scalable multicloud cost optimizer for complex workloads(https://research.redhat.com/blog/article/meet-cco-a-scalable-multicloud-cost-optimizer-for-complex-workloads/)

Ilya Kolchinsky

FEATURE

Tuning Linux kernel policies for energy efficiency with machine learning(https://research.redhat.com/blog/article/tuning-linux-kernel-policies-for-energy-efficiency-with-machine-learning/)

Han Dong

PERSPECTIVES

Research perspectives: Focus on open hardware (https://research.redhat.com/blog/article/research-perspectives-focus-on-open-hardware/)

Ahmed Sanaullah

Ulrich Drepper

FEATURE

Open source education: from philosophy to reality(https://research.redhat.com/blog/article/open-source-education-from-philosophy-to-reality/)

Danni Shi

FEATURE

A data-driven approach for analyzing Common Criteria and FIPS 140 security certificates (https://research.redhat.com/blog/article/a-data-driven-approach-for-analyzing-common-criteria-and-fips-140-security-certificates/)

Jaroslav Řezník

Petr Švenda

PERSPECTIVES

Research perspectives: Focus on clouds and research

IT(https://research.redhat.com/blog/article/research-perspectives-focus-on-clouds-and-research-it/)

Heidi Dempsey

Gagan Kumar

PERSPECTIVES

Research perspectives: Focus on testing and operations(https://research.redhat.com/blog/article/research-perspectives-focus-on-testing-and-operations/)

Bandan Das

Daniel Bristot de Oliveira

Daniel Bristot de Oliveira

PERSPECTIVES

Research perspectives: Focus on security, privacy, and cryptography(https://research.redhat.com/blog/article/research-perspectives-focus-on-security-privacy-and-cryptography/)

Lily Sturmann

PERSPECTIVES

Research perspectives: Focus on AI and machine learning(https://research.redhat.com/blog/article/research-perspectives-focus-on-ai-and-machine-learning/)

Sanjay Arora

Marek Grác

Marek Grac

ABOUT

Red Hat Research connects Red Hat engineers with professors, researchers, and students to bring great research ideas into open source communities. Our activities around the world have produced grants from government and industry, papers at top conferences, and results that have landed in open source projects of all kinds. Red Hat Research welcomes participation from research-minded individuals around the world.

(https://www.facebook.com/RedHatInc) (https://twitter.com/RedHat) (https://www.instagram.com/redhatinc/?hl=en) (https://www.youtube.com/user/RedHatVideos) (https://www.linkedin.com/company/red-hat/)

Red Hat Research

Privacy statement(https://www.redhat.com/en/about/privacy-policy)

Terms of use(https://www.redhat.com/en/about/terms-use)

All policies and guidelines(https://www.redhat.com/en/about/all-policies-guidelines)