AN INFORMAL PUBLICATION FROM ACADEMIA'S PREMIERE STORAGE SYSTEMS RESEARCH CENTER DEVOTED TO ADVANCING THE STATE OF THE ART IN STORAGE AND INFORMATION INFRASTRUCTURES.

CONTENTS

PDL CONSORTIUM MEMBERS

Broadcom, Ltd.
Citadel
Dell EMC
Google
Hewlett-Packard Labs
Hitachi, Ltd.
Intel Corporation
Microsoft Research
MongoDB
NetApp, Inc.
Oracle Corporation
Samsung Information Systems America
Seagate Technology
Tintri
Two Sigma
Toshiba
Veritas
Western Digital

PDL PACKET

NEWSLETTER ON PDL ACTIVITIES AND EVENTS • SPRING 2017
http://www.pdl.cmu.edu/

Automatic Database Management System Tuning Through Large-scale Machine Learning

by Dana Van Aken

Database management systems (DBMSs) are the most important component of any modern data-intensive application. But, this is historically a difficult task because DBMSs have hundreds of configuration "knobs" that control everything in the system, such as the amount of memory to use for caches and how often data is written to storage. This means that organizations often hire human experts to help with tuning activities, though this method can be prohibitively expensive.

OtterTune is a new automatic DBMS configuration tool that overcomes these challenges, making it easier for anyone to deploy a DBMS able to handle large amounts of data and more complex workloads without any expertise needed in database administration. The key feature of OtterTune is that it reduces the amount of time and resources it takes to tune a DBMS for a new application by leveraging knowledge gained from previous DBMS deployments. OtterTune maintains a repository of data collected from previous tuning sessions, and uses this data to build models of how the DBMS responds to different knob configurations. For a new application, these models are used to guide experimentation and recommend optimal settings to the user to improve a target objective (e.g., latency, throughput).

In this article, we first provide a high-level example to demonstrate the tasks performed by each of the components in the OtterTune system, and show how they interact with one another when automatically tuning a DBMS's configuration. We next discuss each of the components in OtterTune's ML pipeline. Finally, we conclude with an evaluation of OtterTune's efficacy by comparing the performance of its best configuration with ones selected by DBAs and other automatic tuning tools for MySQL and Postgres. All the experiments for our evaluation and data collection were conducted on Amazon EC2 Spot Instances. OtterTune's ML algorithms were implemented using Google TensorFlow and Python's scikit-learn.

The OtterTune System

Figure 1 shows an overview of the components in the OtterTune system. At the start of a new tuning session, the DBA tells OtterTune what metric to optimize when selecting a configuration (e.g., latency, throughput). The client-side controller then connects to the target DBMS and collects its Amazon EC2 instance type and current knob configuration.

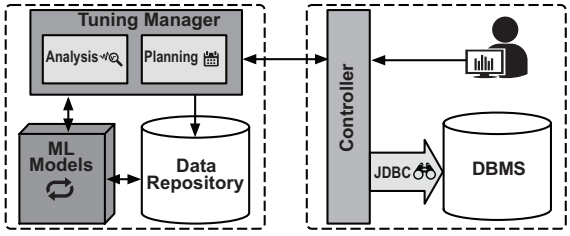


Figure 1: The OtterTune System

Next, the controller starts its first observation period, during which the controller

continued on page 10

FROM THE DIRECTOR'S CHAIR

GREG GANGER

Hello from fabulous Pittsburgh!

It has been another great year for the Parallel Data Lab. Some highlights include record enrollment in PDL's storage systems and cloud classes, continued growth and success for PDL's database systems and big-learning systems research, and new directions for cloud and NVM research. Along the way, many students graduated and joined PDL Consortium companies, new students joined the PDL, and many cool papers have been published. Let me highlight a few things.

Since it headlines this newsletter, I'll also lead off my highlights by talking about PDL's database systems research. The largest scope activities focus on automation in DBMSs, such as in the front-page article, and creating a new open source DBMS called Peloton to embody these and other ideas from the database group. There also continue to be cool results and papers on deduplication in databases, better exploitation of NVM in databases, and improved concurrency control mechanisms. The energy that Andy has infused into database systems research at CMU makes it a lot of fun to watch and participate in.

In addition to applying machine learning (ML) to systems for automation purposes, we continue to explore new approaches to system support for large-scale machine learning. As discussed in the second newsletter article, we have been especially active in exploring how ML systems should adapt to cloud computing environments.

Important questions include how such systems should address dynamic resource availability, unpredictable levels of time-varying resource interference, and geo-distributed data. And, in a fun twist, we are developing new approaches to automatic tuning for ML systems—ML to improve ML.

I continue to be excited about the growth and evolution of the storage systems and cloud classes created and led by PDL faculty… the popularity of both is at an all-time high. For example, over 100 MS students completed the storage class last Fall, and the project-intensive cloud classes serve even more. We refined the new myFTL project such that the students' FTLs store real data in the NAND Flash SSD simulator we provide them, which includes basic interfaces for read-page, write-page, and erase-block. The students write FTL code to maintain LBA-to-physical mappings, perform cleaning, and address wear leveling. In addition to our lectures and the projects, these classes feature 3-5 corporate guest lecturers (thank you, PDL Consortium members!) bringing insight on real-world solutions, trends, and futures.

PDL continues to explore questions of resource scheduling for cloud computing, which grows in complexity as the breadth of application and resource types grows. Our Tetrisched project has developed new ways of allowing users to express their per-job resource type preferences (e.g., machine locality or hardware accelerators) and then exploring the trade-offs among them to maximize utility of the public and/or private cloud infrastructure. Our most recent work explores more

THE PDL PACKET

THE PARALLEL DATA LABORATORY

School of Computer Science
Department of ECE
Carnegie Mellon University
5000 Forbes Avenue
Pittsburgh, PA 15213-3891
VOICE 412•268•6716
FAX 412•268•3010

PUBLISHER

Greg Ganger

EDITOR

Joan Digney

The PDL Packet is published once per year to update members of the PDL Consortium. A pdf version resides in the Publications section of the PDL Web pages and may be freely distributed. Contributions are welcome.

THE PDL LOGO

Skibo Castle and the lands that comprise its estate are located in the Kyle of Sutherland in the northeastern part of Scotland. Both 'Skibo' and 'Sutherland' are names whose roots are from Old Norse, the language spoken by the Vikings who began washing ashore regularly in the late ninth century. The word 'Skibo' fascinates etymologists, who are unable to agree on its original meaning. All agree that 'bo' is the Old Norse for 'land' or 'place,' but they argue whether 'ski' means 'ships' or 'peace' or 'fairy hill.'

Although the earliest version of Skibo seems to be lost in the mists of time, it was most likely some kind of fortified building erected by the Norsemen. The present-day castle was built by a bishop of the Roman Catholic Church. Andrew Carnegie, after making his fortune, bought it in 1898 to serve as his summer home. In 1980, his daughter, Margaret, donated Skibo to a trust that later sold the estate. It is presently being run as a luxury hotel.

FACULTY

Greg Ganger (PDL Director)
412•268•1297
ganger@ece.cmu.edu

David Andersen
Lujo Bauer
Nathan Beckmann
Chuck Cranor
Lorrie Cranor
Christos Faloutsos
Kayvon Fatahalian
Eugene Fink
Rajeev Gandhi
Saugata Ghose
Phil Gibbons
Garth Gibson
Seth Copen Goldstein
Mor Harchol-Balter
Todd Mowry
Onur Mutlu
Priya Narasimhan
David O'Hallaron
Andy Pavlo
Majd Sakr
M. Satyanarayanan
Srinivasan Seshan
Alex Smola
Hui Zhang

STAFF MEMBERS

Bill Courtright, 412•268•5485
(PDL Executive Director) wcourtright@cmu.edu
Karen Lindenfelser, 412•268•6716
(PDL Administrative Manager) karen@ece.cmu.edu
Jason Boles
Joan Digney
Chad Dougherty
Mitch Franzos
Charlene Zang

VISITING RESEARCHERS / POST DOCS

George Amvrosiadis
Daniel Berger
Akira Kuroda
Hyeontaek Lim
Michael (Tieying) Zhang

GRADUATE STUDENTS

Abutalib Aghayev
Joy Arulraj
Rachata Ausavarungnirun
Ben Blum
Amirali Boroumand
Sol Boucher
Christopher Canel
Kevin Chang
Andrew Chung
Henggang Cui
Wei Dai
Debanshu Das
Utsav Drolia
Chris Fallin
Jian Fang
Kristen Scholes Gardner
Kiryong Ha
Aaron Harlap
Kevin Hsieh
Saksham Jain
Angela Jiang
Junchen Jiang
Wesley Jin
Abishek Joshi
Ellango Jothimurugesan
Saurabh Arun Kadekodi
Anuj Kalia
Rajat Kateja
Jin Kyu Kim
Thomas Kim
Dimitris Konomis
Michael Kuchnik
Conglong Li
Mu Li
Yang Li
Haibin Lin
Yixin Luo
Lin Ma
Charles McGuffey
Prashanth Menon
Nathan Mickulicz
Ravi Teja Mullapudi
Jun Woo Park
Akansha Patel
Matt Perron
Alex Poms
Aurick Qiao
Kai Ren
Dana Van Aken
Nandita Vijaykumar
Haoran Wang
Ziqi Wang
Jinliang Wei
Lin Xiao
Pengtao Xie
Hongyi Xin
Lianghong Xu
Jiajun Yao
Yunfan Ye
Huanchen Zhang
Rui Zhang
Qing Zheng
Dong Zhou
Timothy Zhu

FROM THE DIRECTOR'S CHAIR

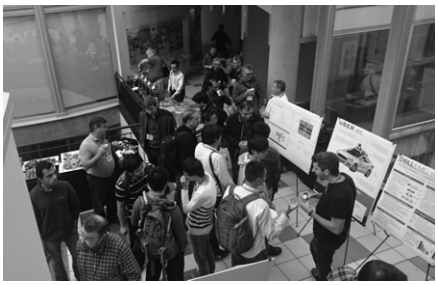
robust ways of exploiting estimated runtime information, finding that providing full distributions of likely runtimes (e.g., based on history of "similar" jobs) works quite well for real-world workloads as reflected in real cluster traces. In collaboration with the new Intel-funded visual cloud systems research center, we are also exploring how processing of visual data should be partitioned and distributed across capture (edge) devices and core cloud resources.

Our explorations of how systems should be designed differently to exploit and work with new underlying storage technologies, especially NVM and SMR, continues to expand on many fronts. Activities range from new search and sort algorithms that understand the asymmetry between read and write performance (i.e., assuming using NVM directly) to a modified Linux Ext4 file system implementation that mitigates drive-managed SMR performance overheads. We're excited about continuing to work with PDL companies on understanding where the hardware is (and should be) going and how it should be exploited in systems.

Naturally, PDL's long-standing focus on scalable storage continues strongly. As always, a primary challenge is metadata scaling, and PDL researchers are exploring several approaches to dealing with scale along different dimensions. A cool new concept being explored is allowing high-end applications to manage their own namespaces, for periods of time, bypassing traditional metadata bottlenecks entirely during the heaviest periods of activity.

Many other ongoing PDL projects are also producing cool results. For example, to help our (and others') file systems research, we have developed a new file system aging suite, called Geriatrix. Our key-value store research continues to expose new approaches to indexing and remote value access. Our continued operation of private clouds in the Data Center Observatory (DCO) serves the dual purposes of providing resources for real users (CMU researchers) and providing us with invaluable Hadoop logs, instrumentation data, and case studies. This newsletter and the PDL website offer more details and additional research highlights.

I'm always overwhelmed by the accomplishments of the PDL students and staff, and it's a pleasure to work with them. As always, their accomplishments point at great things to come.



Industry guests gather with PDL faculty, staff and students in Roberts Hall for breakfast and an industry poster session at the opening of the 2016 PDL Retreat.

Ben Blum shows some details of his work on "Systematic Testing with Data-Race Preemption Points" with Jeff Heller of NetApp at the 2016 PDL Retreat.

YEAR IN REVIEW

May 2017

- 19th annual Spring Visit Day.
- Timothy Zhu defended his PhD thesis "Meeting Tail Latency SLOs in Shared Networked Storage."
- Jinliang Wei presented his speaking skills talk on "Managed Communication and Consistency for Fast Data-Parallel Iterative Analytics."
- Rachata Ausavarungnirun successfully defended his PhD thesis "Techniques for Shared Resource Management in Systems with Graphics Processing Units."
- Presentations at the SIGMOD Int'l Conference on Data Management 2017 included "Automatic Database Management System Tuning Through Large-scale Machine Learning" (Dana Van Aken), and "Online Deduplication for Databases" (Lianghong Xu).
- Rajat Kateja will be interning at Google with their cloud platforms team this coming summer, working with Niket Agarwal.
- Saksham Jain will be interning in VMware in Palo Alto this summer in their VMKernel Team.
- Charles McGuffey will be interning with Google in Mountain View, CA this summer.
- Aaron Harlap will be interning with Microsoft, working on ML for Big Data.

PDL staff members Jason Bole and Charlene Zang discuss PDL infrastructure while enjoying the PDL Retreat.

April 2017

- Henggang Cui successfully defended his PhD research on "Exploiting Application Characteristics for Efficient System Support of Data-Parallel Machine Learning."
- Ben Blum proposed his thesis research "Practical Concurrency Testing."
- Priya Narasimhan honored with Pittsburgh History Makers Award.
- Aaron Harlap presented "Proteus: Agile ML Elasticity through Tiered Reliability in Dynamic Resource Markets" at EuroSys '17 in Belgrade, Serbia.

March 2017

- Presentations at NSDI '17 in Boston, MA included "AdaptSize: Orchestrating the Hot Object Memory Cache in a Content Delivery Network" (Daniel Berger), and "Gaia: Geo-Distributed Machine Learning Approaching LAN Speeds" (Kevin Hsieh).
- Utsav Drolia presented "Towards Edge-caching for Image Recognition" at SmartEdge '17 in Hawaii.

February 2017

- Mu Li successfully defended his PhD thesis "Scaling Distributed Machine Learning with System and Algorithm Co-design."
- Anuj Kalia was named a 2017 Facebook Fellow.
- Abutalib Aghayev presented "Evolving Ext4 for Shingled Disks" at FAST '17 in Santa Clara, CA.
- Two papers from Onur Mutlu's group were presented at HPCA '17 in Austin, TX: "Vulnerabilities in MLC NAND Flash Memory Programming: Experimental Analysis, Exploits, and Mitigation Techniques"
- (Yu Cai), and "SoftMC: A Flexible and Practical Open-Source Infrastructure for Enabling Experimental DRAM Studies" (Hasan Hassan).

January 2017

- Adam Pennington receives CMU Alumni Award.
- Abutalib Aghayev receives Hugo Patterson PDL Entrepreneurship Fellowship.
- Andy Pavlo presented "Self-Driving Database Management Systems" CIDR '17 in Chaminade, CA.

December 2016

- Todd Mowry became an ACM Fellow.

November 2016

- Jiaqi Tan successfully defended his PhD dissertation "Prescriptive Safety-Checks through Automated Proofs for Control-Flow Integrity."
- Jun Woo Park gave his speaking skills talk on "JVuSched: Robust Scheduling with Auto-estimated Job Runtimes."
- Anuj Kalia presented "FaSST: Fast, Scalable and Simple Distributed Transactions with Two-sided (RDMA) Datagram RPCs" at OSDI '17 in Savannah, GA.
- Jiaqi Tan presented "AUSPICE-R: Automatic Safety-Property Proofs for Realistic Features in Machine Code" at ASPLAS '16 in Hanoi, Vietnam.

October 2016

- 24th annual PDL Retreat.
- Ben Blum presented "Stateless Model Checking with Data-Race Preemption Points" at SPLASH 2016 in Amsterdam, Netherlands.
- Kiryong Ha successfully defended his PhD thesis "System Infrastructure for Mobile-Cloud Convergence."
- Mu Li proposed his thesis research "Scaling Distributed Machine Learning with System and Algorithm Co-design."
- Timothy Zhu presented "SNC-Meister: Admitting More Tenants with Tail Latency SLOs" at SoCC '16 in Santa Clara, CA.

continued on page 32

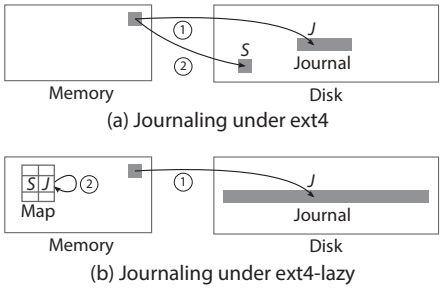
Evolving Ext4 for Shingled Disks

Abutalib Aghayev, Theodore Ts'o, Garth Gibson & Peter Desnoyers

15th USENIX Conference on File and Storage Technologies (FAST '17), Feb 27–Mar 2, 2017. Santa Clara, CA.

Drive-Managed SMR (Shingled Magnetic Recording) disks offer a plug-compatible higher-capacity replacement for conventional disks. For non-sequential workloads, these disks show bimodal behavior: After a short period of high throughput they enter a continuous period of low throughput.

We introduce ext4-lazy, a small change to the Linux ext4 file system that significantly improves the throughput in both modes. We present benchmarks on four different drive-managed SMR disks from two vendors, showing that ext4-lazy achieves 1.7-5.4X improvement over ext4 on a metadata-light file server benchmark. On metadata-heavy benchmarks it achieves 2-13X improvement over ext4 on drive-managed SMR disks as well as on conventional disks.



(a) Ext4 writes a metadata block to disk twice. It first writes the metadata block to the journal at some location J and marks it dirty in memory. Later, the writeback thread writes the same metadata block to its static location S on disk, resulting in a random write. (b) Ext4-lazy, writes the metadata block approximately once to the journal and inserts a mapping (S,J) to an in-memory map so that the file system can find the metadata block in the journal.

Principled Workflow-centric Tracing of Distributed Systems

Raja R. Sambasivan, Ilari Shafer, Jonathan Mace, Benjamin H. Sigelman, Rodrigo Fonseca & Gregory R. Ganger

ACM Symposium on Cloud Computing 2016 Santa Clara, CA,October 5 - 7, 2016.

Workflow-centric tracing captures the workflow of causally-related events (e.g., work done to process a request) within and among the components of a distributed system. As distributed systems grow in scale and complexity, such tracing is becoming a critical tool for understanding distributed system behavior. Yet, there is a fundamental lack of clarity about how such infrastructures should be designed to provide maximum benefit for important management tasks, such as resource accounting and diagnosis. Without research into this important issue, there is a danger that workflow-centric tracing will not reach its full potential. To help, this paper distills the design space of workflow-centric tracing and describes key design choices that can help or hinder a tracing infrastructure's utility for important tasks. Our design space and the design choices we suggest are based on our experiences developing several previous workflow-centric tracing infrastructures.

Design Guidelines for High Performance RDMA Systems

Anuj Kalia, Michael Kaminsky & David G. Andersen

2016 USENIX Annual Technical Conference (USENIX ATC'16), June 2016. Best Student Paper.

Modern RDMA hardware offers the potential for exceptional performance, but design choices including which RDMA operations to use and how to use them significantly affect observed performance. This paper lays out guidelines that can be used by system designers to navigate the RDMA design space. Our guidelines emphasize paying attention to low-level details such as individual PCIe transactions and NIC architecture. We empirically demonstrate how these guidelines can be used to improve the performance of RDMA-based systems: we design a networked sequencer that outperforms an existing design by 50x, and improve the CPU efficiency of a prior high-performance key-value store by 83%. We also present and evaluate several new RDMA optimizations and pitfalls, and discuss how they affect the design of RDMA systems.

SNC-Meister: Admitting More Tenants with Tail Latency SLOs

Timothy Zhu, Daniel S. Berger & Mor Harchol-Balter

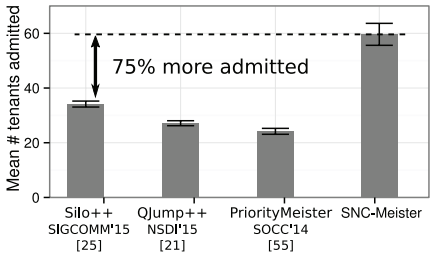
ACM Symposium on Cloud Computing 2016 (SoCC'16), Santa Clara, CA, October 5 - 7, 2016.

Meeting tail latency Service Level Objectives (SLOs) in shared cloud networks is both important and challenging. One primary challenge is determining limits on the multitenancy such that SLOs are met. Doing so involves estimating latency, which is difficult, especially when tenants exhibit bursty behavior as is common in production environments. Nevertheless, recent papers in the past two years (Silo, QJump, and PriorityMeister) show techniques for calculating latency based on a branch of mathematical modeling called Deterministic Network Calculus (DNC). The DNC theory is designed for adversarial worst-case conditions, which is sometimes necessary, but is often overly conservative. Typical tenants do not require strict worst-case guarantees, but are only looking for SLOs at lower percentiles (e.g., 99th, 99.9th).

This paper describes SNC-Meister, a new admission control system for tail latency SLOs. SNC-Meister improves

continued on page 6

continued from page 5



Admission numbers for state-of-the-art admission control systems and SNC-Meister in 100 randomized experiments. In each experiment, 180 tenants, each submitting hundreds of thousands of requests, arrive in random order and seek a 99.9% SLO randomly drawn from f10ms, 20ms, 50ms, 100msg. While all systems meet all SLOs, SNC-Meister is able to support on average 75% more tenants with tail latency SLOs than the next-best system.

upon the state-of-the-art DNC-based systems by using a new theory, Stochastic Network Calculus (SNC), which is designed for tail latency percentiles. Focusing on tail latency percentiles, rather than the adversarial worst-case DNC latency, allows SNC-Meister to pack together many more tenants: in experiments with production traces, SNC-Meister supports 75% more tenants than the state-of-the-art.

Parallel Algorithms for Asymmetric Read-Write Costs

Naama Ben-David, Guy E. Blelloch, Jeremy T. Fineman, Phillip B. Gibbons, Yan Gu, Charles McGuffey & Julian Shun

SPAA 2016. 28th ACM Symposium on Parallelism in Algorithms and Architectures, July 11 - 13, 2016. Asilomar State Beach, CA, USA.

Motivated by the significantly higher cost of writing than reading in emerging memory technologies, we consider parallel algorithm design under such asymmetric read-write costs, with the goal of reducing the number of writes while preserving work-efficiency and low span. We present a nested-parallel model of computation that combines (i) small per-task stack-allocated memories with symmetric read-write costs and (ii) an unbounded heap-allocated shared memory with asymmetric read-write costs, and show how the costs in the model map efficiently onto a more concrete machine model under a work-stealing scheduler. We use the new model to design reduced-write, work-efficient, low-span parallel algorithms for a number of fundamental problems such as reduce, list contraction, tree contraction, breadth-first search, ordered filter, and planar convex hull. For the latter two problems, our algorithms are output-sensitive in that the work and number of writes decrease with the output size. We also present a reduced-write, low-span minimum spanning tree algorithm that is nearly work-efficient (off by the inverse Ackermann function). Our algorithms reveal several interesting techniques for significantly reducing shared memory writes in parallel algorithms without asymptotically increasing the number of shared memory reads.

Cachier: Edge-caching for Recognition Applications

Utsav Drolia, Katherine Guo, Jiaqi Tan, Rajeev Gandhi & Priya Narasimhan

The 37th IEEE International Conference on Distributed Computing Systems (ICDCS 2017), June 5 – 8, 2017, Atlanta, GA, USA.

Recognition and perception-based mobile applications, such as image recognition, are on the rise. These applications recognize the user's surroundings and augment it with information and/or media. These applications are latency-sensitive. They have a soft-realtime nature - late results are potentially meaningless. On the one hand, given the compute-intensive nature of the tasks performed by such applications, execution is typically offloaded to the cloud. On the other hand, offloading such applications to the cloud incurs network latency, which can increase the user-perceived latency. Consequently, edge-computing has been proposed to let devices offload intensive tasks to edge-servers instead of the cloud, to reduce latency. In this paper, we propose a different model for using edge-servers. We propose to use the edge as a specialized cache for recognition applications and formulate the expected latency for such a cache. We show that using an edge-server like a typical web-cache, for recognition applications, can lead to higher latencies. We propose Cachier, a system that uses the caching model along with novel optimizations to minimize latency by adaptively balancing load between the edge and the cloud by leveraging spatiotemporal locality of requests, using offline analysis of applications, and online estimates of network conditions. We evaluate Cachier for image-recognition applications and show that our techniques yield 3x speed-up in responsiveness, and perform accurately over a range of operating conditions. To the best of our knowledge, this is the first work that models edge-servers as caches for compute-intensive recognition applications, and Cachier is the first system that uses this model to minimize latency for these applications.

Automatic Database Management System Tuning Through Large-scale Machine Learning

Dana Van Aken, Andrew Pavlo, Geoffrey J. Gordon & Bohan Zhang

ACM SIGMOD International Conference on Management of Data, May 14-19, 2017. Chicago, IL, USA.

Database management system (DBMS) configuration tuning is an essential aspect of any data-intensive application effort. But this is historically a difficult task because DBMSs have hundreds of configuration "knobs" that control everything in the system, such as the amount of memory to use for caches and how often data is written to storage. The problem with these

continued on page 7

continued from page 6

knobs is that they are not standardized (i.e., two DBMSs use a different name for the same knob), not independent (i.e., changing one knob can impact others), and not universal (i.e., what works for one application may be sub-optimal for another). Worse, information about the effects of the knobs typically comes only from (expensive) experience.

To overcome these challenges, we present an automated approach that leverages past experience and collects new information to tune DBMS configurations: we use a combination of supervised and unsupervised machine learning methods to (1) select the most impactful knobs, (2) map unseen database workloads to previous workloads from which we can transfer experience, and (3) recommend knob settings. We implemented our techniques in a new tool called OtterTune and tested it on three DBMSs. Our evaluation shows that OtterTune recommends configurations that are as good as or better than ones generated by existing tools or a human expert.

Online Deduplication for Databases

Lianghong Xu, Andrew Pavlo, Sudipta Sengupta & Gregory R. Ganger

ACM SIGMOD International Conference on Management of Data, May 14-19, 2017. Chicago, IL, USA.

dbDedup is a similarity-based deduplication scheme for on-line database management systems (DBMSs). Beyond block-level compression of individual database pages or operation log (oplog) messages, as used in today's DBMSs, dbDedup uses byte-level delta encoding of individual records within the database to achieve greater savings. dbDedup's single-pass encoding method can be integrated into the storage and logging components of a DBMS to provide two benefits: (1) reduced size of data stored on disk beyond what traditional compression schemes provide, and (2) reduced amount of data transmitted over the network for replication services. To evaluate our work, we implemented dbDedup in a distributed NoSQL DBMS and analyzed its properties using four real datasets. Our results show that dbDedup achieves up to 37X reduction in the storage size and replication traffic of the database on its own and up to 61X reduction when paired with the DBMS's blocklevel compression. dbDedup provides both benefits with negligible effect on DBMS throughput or client latency (average and tail).

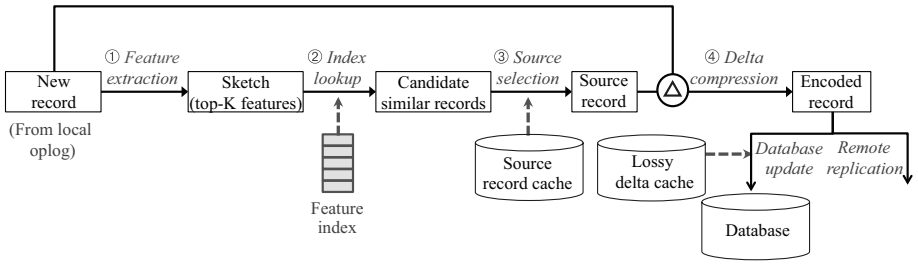
AdaptSize: Orchestrating the Hot Object Memory Cache in a Content Delivery Network

Daniel S. Berger, Ramesh K. Sitaraman & Mor Harchol-Balter

14th USENIX Symposium on Networked Systems Design and Implementation (NSDI '17). March 27–29, 2017, Boston, MA.

Most major content providers use content delivery networks (CDNs) to serve web and video content to their users. A CDN is a large distributed system of servers that caches and delivers content to users. The first-level cache in a CDN server is the memory-resident Hot Object Cache (HOC). A major goal of a CDN is to maximize the object hit ratio (OHR) of its HOCs. But, the small size of the HOC, the huge variance in the requested object sizes, and the diversity of request patterns make this goal challenging.

We propose AdaptSize, the first adaptive, size-aware cache admission policy for HOCs that achieves a high OHR, even when object size distributions and request characteristics vary significantly over time. At the core of AdaptSize is a novel Markov cache model that seamlessly adapts the caching parameters to the changing request patterns. Using request traces from one of the largest CDNs in the world, we show that our implementation of AdaptSize achieves significantly higher OHR than widely-used production systems: 30-48% and 47-91% higher OHR than Nginx and Varnish, respectively. AdaptSize also achieves 33-46% higher OHR than state-of-the-art research systems. Further, AdaptSize is more robust to changing request patterns than the traditional tuning approach of hill climbing and shadow queues studied in other contexts.



dbDedup Workflow – (1) Feature Extraction, (2) Index Lookup, (3) Source Selection, and (4) Delta Compression.

Improving the Reliability of Chip-off Forensic Analysis of NAND Flash Memory Devices

Aya Fukami, Saugata Ghose, Yixin Luo, Yu Cai & Onur Mutlu

DFRWS Digital Forensics Research Conference Europe (DFRWS EU), March 21 - 23, 2017 Lake Constance, Germany.

Digital forensic investigators often need to extract data from a seized device that contains NAND flash memory. Many such devices are physically damaged, preventing investigators from using automated techniques

continued on page 20

AWARDS & OTHER PDL NEWS

April 2017
Priya Narasimhan Honored with History Makers Award

Each year, the History Makers Award Dinner honors five men and women who have made a difference through their work to not only the Greater Pittsburgh area, but beyond. Congratulations to Priya, Professor of Electrical and Computer Engineering, and the CEO and founder of YinzCam who will receive this award at a banquet on May 12 for her outstanding achievements in technology and innovation. Other awards are made in the fields of education, sports, healthcare and community.

-- information from heinzhistorycenter.org

January 2017
Jun Woo Park and Sun Hee Baik Marry!

Congratulations to Jun Woo and Sun Hee, who were married on January 7th, 2017 at home in Seoul, Korea!

January 2017
Anuj Kalia Named a 2017 Facebook Fellow

Congratulations to Anuj Kalia, who has been named to the 2017 class of Facebook fellows!

Founded in 2010, the Facebook Fellowship program is designed to help foster ties with the academic community, encourage and support promising Ph.D. students engaged in research across computer science and engineering, and provide those students with opportunities to work with Facebook on problems relevant to their research. Since its inception, the program has supported more than 50 Ph.D. candidates, whose research covers topics that range from power systems and microgrids to the intersection of computer vision, machine learning and cognitive science.

Anuj is a Ph.D. student in the Computer Science Department, a member of the PDL, and an advisee of Associate Professor David Anderson. He received his B.Tech in computer science from the Indian Institute of Technology-Dehli, and his research interests include networked systems, with a focus on designing efficient software systems for high-speed networks.

-- information from CMU SCS News Jan 31, 2017, Aisha Rashid

January 2017
Adam Pennington Receives CMU Alumni Award

First presented in 1950, the Carnegie Mellon Alumni Awards recognize alumni, students, and faculty for their service to the university and its alumni, for their achievements in the arts, humanities, sciences, technology, and business. To date more than 880 individuals have been honored through the program.

Congratulations to Adam (CS 2001, E 2003), who has been awarded a 2016 CMU Alumni Award for Alumni Service. He has been involved with various Carnegie Mellon University volunteer roles since he graduated with a BS in Computer Science in 2001, and an MS in Electrical and Computer Engineering in 2003. He is the current president of the Activities Board Technical Committee (ABTech) alumni interest group, which he founded in 2004. In 2015, Adam started the ABTech Advancement Fund, which is dedicated to helping current undergraduate members of ABTech attend industry conferences. Adam also served on the Alumni Association Board from 2011 until 2015, co-chairing both the Networks and Awards committees.

While at CMU, Adam was an executive in ABTech, a board member of Scotch 'n' Soda, and an officer of both Hammerschlag tower clubs (Carnegie Tech Radio Club [W3VC] and the ECE Graduate Organization). He currently works for a nonprofit and spends as much time as possible SCUBA diving with his girlfriend, Lindsay Spriggs (HS 2005, 2006).

-- information from CMU Alumni Association News, January 2017

January 2017
Hugo Patterson PDL Entrepreneurship Fellowship Awarded

The Parallel Data Laboratory is pleased to announce that the Hugo Patterson PDL Entrepreneurship Fellowship has been Awarded to Abutalib Aghayev. This award provides academic (tuition

continued on page 9

continued from page 8

and stipend) support to a student working within the umbrella of the PDL who has had experience in the workforce prior to entering a graduate program.

Abutalib Aghayev is a Ph.D. student in the Computer Science Department at Carnegie Mellon University. He received his M.S. degree from Northeastern University and B.S. degree from Bogazici University, Istanbul, Turkey. Before coming to United States for graduate studies, he was a co-founder and CTO of a software startup in Istanbul. His research interests are in storage and systems support for machine learning.

The endowment for the fellowship was made by PDL alum Hugo Patterson, an entrepreneur who wishes to foster similar opportunities for PDL graduate students. He is currently the CTO and Co-Founder of Datrium, where he has helped develop a Server Flash Storage System for private clouds. It's a new kind of storage that leverages server cycles and server flash and combines them with an external NetShelf for durable storage. Prior to Datrium, Hugo was an EMC Fellow serving as CTO of the EMC Backup Recovery Systems Division, and the Chief Architect and CTO of Data Domain (acquired by EMC in 2009), where he built the first deduplication storage system. Prior to that he was the engineering lead at NetApp, developing SnapVault, the first snap-and-replicate disk-based backup product. Hugo holds a Ph.D. in Electrical and Computer Engineering from Carnegie Mellon University, and was the inaugural recipient of the CMU Parallel Data Lab Distinguished Alumni Award.

December 2016
Todd Mowry Named ACM Fellow

Congratulations to Todd Mowry on being named an ACM Fellow for 2016! Mowry, a professor in the Computer Science Department, was cited "for contributions to software prefetching and thread-level speculation." His research interests span the areas of computer architecture, compilers, operating systems, parallel processing, database performance, and modular robotics. He co-leads the Log-Based Architectures project and the Claytronics project. In 2004-2007, he served as the director of Intel Labs Pittsburgh. Todd was among 53 members of the ACM, the world's leading computing society, elevated to fellow status this year. ACM will formally recognize the 2016 fellows at the annual Awards Banquet in San Francisco on June 24, 2017.

-- Carnegie Mellon University News, Dec. 8, 2016, Byron Spice

July 2016
Announcing the Carnegie Mellon Database Application Catalog

The Carnegie Mellon Database Application Catalog (CMDBAC) is an on-line repository of open-source database applications that you can use for benchmarking and experimentation. The goal of this project is to provide ready-to-run real-world applications for researchers and practitioners that go beyond the standard benchmarks.

We built a crawler that finds applications hosted on public repositories (e.g., GitHub). We then created a framework that automatically learns how to deploy and execute an application inside a virtual machine sandbox. You can then safely download the application on your local machine and execute it to collect query traces and other metrics.

The CMDBAC currently contains over 1000 applications of varying complexity. Web applications based on popular programming frameworks are targeted because (1) they are easier to find and (2) we can automate the deployment process. We support applications that use the Django, Ruby on Rails, Drupal, Node.js, and Grails frameworks. Find the CMDBAC website at: http://cmdbac.cs.cmu.edu and the source code at https://github.com/cmu-db/cmdbac.

June 2016
Best Student Paper at ATC'16!

Congratulations to Anuj Kalia and his co-authors Michael Kaminsky and David G. Andersen for receiving the award for Best Student Paper at the 2016 USENIX Annual Technical Conference (USENIX ATC'16), in Denver, CO. Their paper, "Design Guidelines for High Performance RDMA Systems," lays out guidelines that can be used by system designers to navigate the RDMA design space in order to take advantage of potential performance improvements.

continued from page 1

observes the DBMS and measures DBMS-independent external metrics chosen by the DBA, such as the latency and throughput. At the end of the observation period, the controller collects additional DBMS-specific internal metrics, such as MySQL's counters for pages read from or written to disk. The controller then returns both the external and internal metrics to the tuning manager.

When OtterTune's tuning manager receives the result of a new observation period from the controller, it first stores that information in its repository. From this, OtterTune computes the next configuration that the controller should install on the target DBMS. This configuration, as well as an estimate of the expected improvement to be gained by running this configuration, are returned to the controller. The estimate helps the DBA decide whether to iterate or terminate the tuning session.

We note that there are certain assumptions made by the OtterTune system that may limit its capabilities for some users. For example, it assumes that the DBA has administrative privileges so that the controller can modify the DBMS's configuration. A complete discussion of these assumptions and limitations is provided in our paper [1].

Machine Learning Pipeline

Figure 2 shows the processing path of data as it moves through OtterTune's ML pipeline. All previous observations reside in the OtterTune repository. This data is first passed into the *workload characterization* component that identifies the most distinguishing DBMS metrics. Next, the *knob identification* component generates a ranked list of the most important knobs. This information is then fed to the *automatic tuner* component where it maps the target DBMS's workload to a previously seen workload and generates improved tuning configurations.

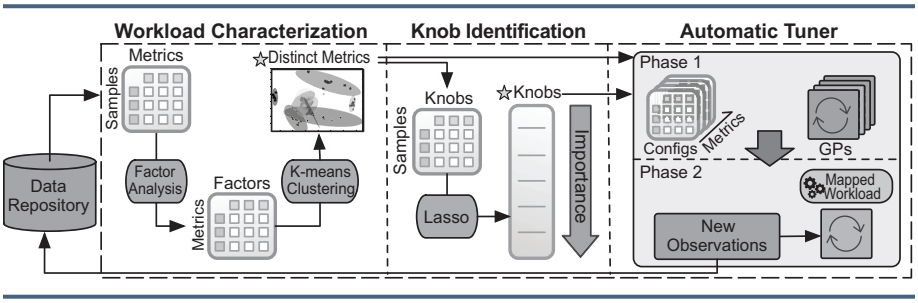


Figure 2: OtterTune Machine Learning Pipeline.

Following is a discussion of the components in the ML pipeline in more detail.

Workload Characterization: OtterTune uses the DBMS's internal runtime metrics to characterize how a workload behaves. These metrics provide an accurate representation of a workload because they capture many aspects of its runtime behavior. The problem with these metrics is that many of them are redundant: some are the same measurement recorded in different units; others are ones that represent independent components of the DBMS, but whose values are highly correlated. Pruning these redundant metrics is important to reduce the complexity of the ML models that use them.

Knob Identification: The knob identification component determines which knobs have the strongest impact on the DBA's target objective function. DBMSs can have hundreds of knobs, but only a subset affect the DBMS's performance. OtterTune uses a popular feature selection technique for linear regression, called Lasso, to identify important knobs and the dependencies that may exist between them. Applying this technique to the knob data and the data for the non-redundant set of metrics in OtterTune's repository exposes the knobs that have the strongest correlation to the system's overall performance.

Automatic Tuner: The automated tuning component executes a two-step analysis after the completion of each observation period in the tuning process to determine which configuration it should recommend next.

In the first step, the system identifies which workload from a previous tuning session is most representative of the target workload based on the performance measurements for the non-redundant set of metrics. It does this by comparing the session's metrics with those from previously seen workloads to see which ones react similarly to different knob settings.

In the next step, OtterTune chooses a configuration that is explicitly selected to maximize the target objective over the entire tuning session. Instead of tuning all of the DBMS's configuration knobs, OtterTune selects only the top-k from its ranked list of knobs to tune. OtterTune reuses the data from the most similar workload in its repository and the data collected so far from the target workload (for the top-k knobs and the target metric) to train a Gaussian Process model. This model enables OtterTune to estimate how well the target metric will perform for a given DBMS configuration.

Experimental Evaluation

OtterTune's system is written in Python. The ML algorithms used in the Workload Characterization and Knob Identification components are implemented using scikit-learn. These algorithms are frequently recomputed by background processes as new data becomes available in OtterTune's repository. For instance, the algorithms from the Automatic Tuning component are recomputed after each observation period. We implemented these algorithms using

continued on page 11

continued from page 10

TensorFlow as performance is more of a consideration here. We integrated OtterTune's controller with a benchmarking framework, called OLTP-Bench, to collect data about the DBMS's hardware, knob configurations, and runtime performance metrics.

We conducted all our experiments on Amazon EC2. Each experiment consists of two instances: OtterTune's controller and the target DBMS deployment. For these, we used the m4.large and m3.xlarge instance types, respectively. We deployed OtterTune's tuning manager and repository on a local server with 20 cores and 128GB RAM.

Our evaluation compares the performance of MySQL and Postgres when using the best configuration selected by OtterTune versus ones selected by human DBAs and open-source tuning advisor tools. We also compare OtterTune's configurations with those created for Amazon RDS-managed DBMSs deployed on the same EC2 instance type as the rest of our experiments.

For this comparison, we use the TPC-C workload, which is the current industry standard for evaluating the performance of OLTP systems.

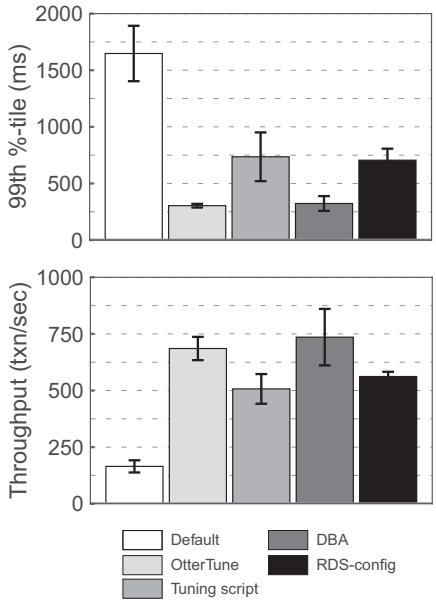


Figure 3: Efficacy Comparison (MySQL).

MySQL

The results in Figure 3 show that MySQL achieves 22-35% better throughput and approximately 60% better latency when using the best configuration generated by OtterTune versus ones generated by the tuning script and Amazon RDS for the TPC-C workload. We also see that OtterTune generates a configuration that is almost as good as the DBA.

We find that MySQL has a small number of knobs that have a large impact on its performance for the TPC-C workload. The configurations generated by OtterTune and the DBA provided good settings for each of these knobs. Amazon RDS performs slightly worse because it provides a sub-optimal setting for one of the knobs. The tuning script's configuration performs the worst because it only modifies one knob.

Postgres

The latency measurements in our results show that the configurations generated by OtterTune, the tuning tool, the DBA, and RDS all achieve similar improvements for TPC-C over Postgres' default settings. This is likely because of the overhead of network round-trips between the OLTP-Bench client and the DBMS. But the throughput measurements show that Postgres has ~12% higher performance with OtterTune compared to the DBA and the tuning script, and ~32% higher performance compared to RDS.

Similar to MySQL, there are only a few knobs that impact Postgres' performance for the TPC-C workload. But in this case, the configurations generated by OtterTune, the DBA, the tuning script and Amazon RDS all modified these knobs and most provided reasonably good settings.

Conclusion

OtterTune is an automatic DBMS configuration tool that reuses training data gathered from previous tuning sessions to tune new DBMS deployments. This drastically reduces tuning time as OtterTune does not need to generate an initial data set for training its ML models.

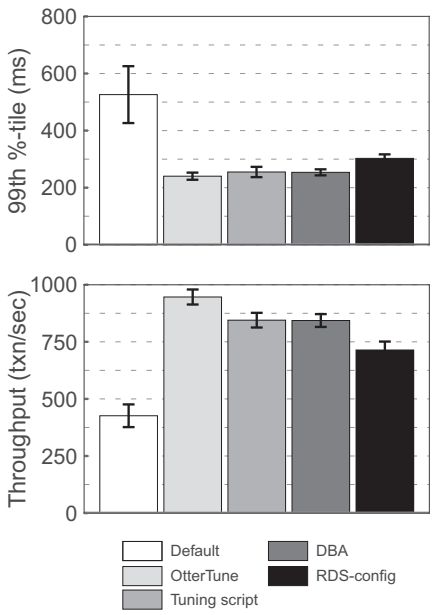


Figure 4: Efficacy Comparison (Postgres).

Our next step in this research is to enable OtterTune to automatically detect the hardware capabilities of the target DBMS without having remote access to the DBMS's host machine. This restriction is necessary due to the growing popularity of DBaaS deployments where such access is not available.

For more details about the OtterTune system, take a look at our paper [1] or check out the code [2] on Github. We are currently building a website to provide OtterTune as an online-tuning service.

References

[1] Automatic Database Management System Tuning Through Large-scale Machine Learning. Dana Van Aken, Andrew Pavlo, Geoffrey J. Gordon, Bohan Zhang. ACM SIGMOD International Conference on Management of Data, May 14-19, 2017. Chicago, IL.

[2] https://github.com/cmu-db/ottertune

BIG-LEARNING SYSTEMS MEET CLOUD COMPUTING

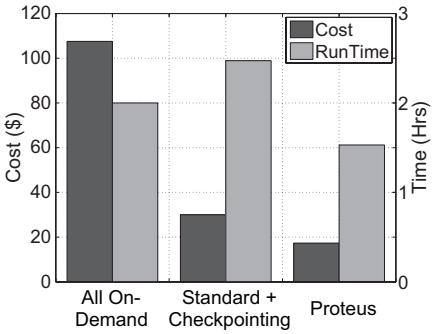
Greg Ganger and PDL Big-learning Group

Statistical machine learning (ML) has become a primary data processing activity for business, science, and online services that attempt to extract insight from observation (training) data. Generally speaking, ML algorithms iteratively process training data to determine model parameter values that make an expert-chosen statistical model best fit it. Once fit (trained), such models can predict outcomes for new data items based on selected characteristics (e.g., for recommendation systems), correlate effects with causes (e.g., for genomic analyses of diseases), label new media files (e.g., which ones are funny cat videos?), and so on.

Given ever-growing quantities of data being collected and stored, combined with ubiquitous attempts to discover and employ new ways of leveraging it, it is not surprising that there is work on ML systems at every scale. PDL researchers have been at the forefront of systems support for large-scale ML, which we refer to as "big-learning systems," and exciting new questions have arisen around how best to adapt to and exploit properties of cloud computing. The convergence of big-learning systems and cloud computing brings new challenges, like the increased straggler effects addressed in our FlexRR work, and many new opportunities. This short article highlights two of our newest works in this space, focused on exploiting transient resource availability and on big-learning for geo-distributed data, respectively.

Proteus: Agile ML Elasticity for Dynamic Resource Markets

ML model training is often quite resource intensive, requiring hours on 10s or 100s of cores to converge on a solution. As such, it should be able to exploit any available extra resources or cost savings. Many modern compute infrastructures offer a great opportunity: transient availability of cheap but revocable resources. For example, Amazon EC2's spot market and Google Compute Engine's preemptible instances often allow customers to use machines at a 70–80% discount off the regular price, but with the risk that they can be taken away at any time. Similarly, many cluster schedulers allow lower-priority jobs to use resources provisioned but not currently needed to support business-critical activities, taking the resources away when those activities need them. ML model training could often be faster and/or cheaper by aggressively exploiting such revocable resources.



Cost and time benefits of Proteus. This graph shows average cost (left axis) and runtime (right axis) for running the MLR application on the AWS EC2 US-EAST-1 Region. The three configurations shown are: 128 on-demand machines, using 128 spot market machines with checkpoint/restart for dealing with evictions and a standard strategy of bidding the on-demand price, and Proteus using 3 on-demand and up to 189 spot market machines. Proteus reduces cost by 85% relative to using all on-demand machines and by ≈50% relative to the checkpointing-based scheme.

Unfortunately, efficient modern frameworks for parallel ML, such as TensorFlow, MxNet, and Petuum, are not designed to exploit transient resources. Most use a *parameter server* architecture, in which parallel workers process training data independently and use a specialized key-value store for shared state, offloading communication and synchronization challenges from ML app writers. Like MPI-based HPC applications, these frameworks generally assume that the set of machines is fixed, optimizing aggressively for the no machine failure and no machine change case (and restarting the entire computation from the last checkpoint on any failure). So, using revocable machines risks significant rollback overhead, and adding newly available machines to a running computation is often not supported.

Proteus is a new parameter server system that combines agile elasticity with aggressive acquisition strategies to exploit transient revocable resources. As one example of how well this works, we find that using three on-demand instances and up to 189 spot market instances allows Proteus to reduce cost by 85% when compared to using only on-demand instances, even when accounting for spot market variation and revocations, while running 24% faster.

Proteus consists of two principal components: AgileML and BidBrain. The AgileML parameter-server system achieves agile elasticity by explicitly combining multiple reliability tiers, with core functionality residing on more reliable resources (non-transient resources, like on-demand instances on EC2) with most work performed on transient resources. This allows quick and efficient scaling, including expansion when resources become available and bulk extraction of revoked transient resources without big delays for rolling back state or recovering lost work. AgileML transitions among different modes/stages as transient resources come and go. When the ratio of transient to non-transient is small (e.g., 2-to-1), it simply distributes the parameter server functionality across only the non-transient machines, instead of across all machines, as is the usual approach. For much larger ratios

continued on page 13

continued from page 12

(e.g., 63-to-1), the one non-transient machine would be a bottleneck in that configuration. In that case, AgileML uses non-transient machine(s) as on-line backup parameter servers to active primary parameter servers that run on transient machines. Updates are coalesced and streamed from actives to backups in the background at a rate that the network bandwidth can accommodate.

BidBrain is Proteus's resource allocation component that decides when to acquire and yield transient resources. BidBrain is specialized for EC2, exploiting spot market characteristics in its policies, but its general approach could apply to other environments with transient resources (e.g., private clusters or other cloud costing models). It monitors current market prices for multiple instance types, which move relatively independently, and bids on new resources when their addition to the current footprint would increase work per dollar. Similarly, resources near the end of an hour may be released if they have become less cost-effective relative to others. As part of its considerations, BidBrain estimates the probability of getting free compute cycles due to instance revocation within the billing hour (with later in the hour being better than earlier) for different bids and spot market conditions. Simultaneously considering costs (e.g., revocation and scaling inefficiencies) and benefits (e.g., cheaper new resources), BidBrain finds a happy medium between aggressive bidding on transient resources and more conservative choices.

Experiments with three real ML applications confirm that Proteus achieves significant cost and runtime reductions. AgileML's elasticity support introduces negligible performance overhead, scales well, and suffers minimal disruption during bulk addition or removal of transient machines. In breaking down the sources of benefit, we find that both the agile elasticity of AgileML and the aggressive policies of BidBrain are needed; using either one alone (e.g., BidBrain with checkpointing instead of AgileML) achieves half or less the cost and runtime savings.

For more information, see [1].

Gaia: Geo-distributed ML Approaching LAN Speeds

Machine learning (ML) is widely used to derive useful information from large-scale data (such as user activities, pictures, and videos) generated at increasingly rapid rates, all over the world. Unfortunately, it is often infeasible to move all this globally-generated data to a centralized data center before running an ML algorithm over it; moving large amounts of raw data over wide-area networks (WANs) can be extremely slow, and is also subject to the constraints of privacy and data sovereignty laws. This motivates the need for geo-distributed ML systems that can run on data spanning multiple data centers.

Today's large-scale distributed ML systems are designed and perform well only for operation within a single data center. Our experiments using three state-of-the-art distributed ML systems (Bösen, IterStore, and GeePS) show that operating these systems across as few as two data centers (over WANs) can cause a slowdown of 1.8–53.7× relative to their performance within a data center (over LANs), as the communication overwhelms the limited WAN bandwidth. Existing systems that do address challenges in geo-distributed data analytics focus on bulk data querying rather than the sophisticated ML algorithms commonly run on big-learning systems.

Gaia is a new ML system for geo-distributed data that (1) employs an intelligent communication mechanism over WANs to efficiently utilize scarce WAN bandwidth, while keeping the global ML model sufficiently consistent across data centers to retain the accuracy and correctness guarantees of an ML algorithm; and (2) is generic and flexible enough to run a wide range of ML algorithms, without requiring any changes to the algorithms themselves.

Gaia builds on the widely used parameter server architecture that provides ML worker machines with a distributed global shared memory abstraction for the ML model parameters they collectively train until convergence to fit the input data. The key idea of Gaia is to maintain an approximately-correct copy of the global ML model within each data center, and dynamically eliminate any unnecessary communication between data centers. Gaia enables this by decoupling the synchronization (i.e., communication/consistency) model within a data center from the synchronization model between different data centers. This differentiation allows Gaia to run a conventional synchronization model that maximizes utilization of the more-freely-available LAN bandwidth within a data center.

At the same time, across different data centers, Gaia employs a new synchronization model, the Approximate Synchronous Parallel (ASP) model, which makes more efficient use of the scarce and heterogeneous WAN bandwidth. By ensuring that each ML model copy that exists in different data centers is approximately correct based on a precise notion, ASP maintains ML algorithm convergence guarantees.

ASP is based on a key finding that the vast majority of updates to the global ML model parameters from each ML worker machine are insignificant. For example, our study of three classes of ML algorithms shows that more than 95% of the updates produce less than a

continued on page 13

DISSERTATION ABSTRACT: Meeting Tail Latency SLOs in Shared Networked Storage

Timothy Zhu
Carnegie Mellon University, SCS

PhD Defense — May 3, 2017

Shared computing infrastructures (e.g., cloud computing, enterprise datacenters) have become the norm today due to their lower operational costs and IT management costs. However, resource sharing introduces challenges in controlling performance for each of the workloads using the infrastructure. For user-facing workloads (e.g., web server, email server), one of the most important performance metrics companies want to control is tail latency, the time it takes to complete the most delayed requests. Ideally, companies would be able to specify tail latency performance goals, also called Service Level Objectives (SLOs), to ensure that almost all requests complete quickly.

Meeting tail latency SLOs is challenging for multiple reasons. First, tail latency is significantly affected by the burstiness that is commonly exhibited by production workloads. Burstiness leads to transient queueing, which is a major cause of high tail latency. Second, tail latency is often due to I/O (e.g., storage, networks), and I/O devices exhibit performance peculiarities that make it hard to meet SLOs. Third, the end-to-end latency is affected by sum of latencies across multiple types of resources such as storage and networks. Most of the existing research, however, have ignored burstiness and focused on a single resource.

This thesis introduces new techniques for meeting end-to-end tail latency SLOs in both storage and networks while accounting for the burstiness that arises in production workloads. We address open questions in scheduling policies, admission control, and workload placement. We build a new Quality of Service (QoS) system for meeting tail latency SLOs in networked storage infrastructures. Our system uses prioritization and rate limiting as tools for controlling the congestion between workloads. We introduce a novel approach for intelligently configuring the workload priorities and rate limits using two different types of queueing analyses: Deterministic Network Calculus (DNC) and Stochastic Network Calculus (SNC). By integrating these mathematical analyses into our system, we are able to build better algorithms for optimizing the resource usage. Our implementation results using realistic workload traces on a physical cluster demonstrate that our approach can meet tail latency SLOs while achieving better resource utilization than the state-of-the-art.

While this thesis focuses on scheduling policies, admission control, and workload placement in storage and networks, the ideas presented in our work can be applied to other related problems such as workload migration and datacenter provisioning. Our theoretically grounded techniques for controlling tail latency can also be extended beyond storage and networks to other contexts such as the CPU, cache, etc. For example, in real-time CPU scheduling contexts, our DNC-based techniques could be used to provide strict latency guarantees while accounting for workload burstiness.

DISSERTATION ABSTRACT: Techniques for Shared Resource Management in Systems with Graphics Processing Units

Rachata Ausavarungnirun
Carnegie Mellon University, ECE

PhD Defense — May 2, 2017

The continued growth of the computational capabilities, along with considerable improvements in the programmability of Graphics Processing Units (GPUs), have made GPUs the platform of choice for a wide variety of applications, ranging from typical graphics applications to general purpose data parallel (GPGPU) applications. However, this success has been accompanied by new performance bottlenecks throughout the memory hierarchy of GPU-based systems. This dissertation identifies and eliminates such performance bottlenecks caused by interference throughout the memory hierarchy.

Specifically, we provide an in-depth analysis of inter- and intra-application as well as inter-address-space interference that significantly degrade the performance and efficiency of GPU-based systems.

To minimize such interference, we introduce changes to the memory hierarchy for systems with GPUs that allow the memory hierarchy to be aware of both CPU and GPU applications' memory access characteristics. We introduce mechanisms to dynamically analyze different applications' characteristics and propose four major changes throughout the memory hierarchy.

First, we introduce Memory Divergence Correction (MeDiC), a cache management mechanism that mitigates intra-application interference in GPGPU applications by allowing the shared L2 cache and the memory controller to be aware of the GPU's warp-level memory divergence characteristics. MeDiC uses this warp-level memory divergence information to give more cache and memory bandwidth resources to warps that benefit most from utilizing such resources. Our evaluations show that MeDiC significantly outperforms multiple state-of-the-art caching policies proposed for GPUs.

Second, we introduce the Staged Memory Scheduler (SMS), an application-aware CPU-GPU memory request scheduler that mitigates inter-application interference in heterogeneous CPU-GPU systems. SMS creates a fundamentally new approach

continued on page 15

continued from page 14

to memory controller design that decouples the memory controller into three significantly simpler structures, each of which has a separate task and all of which operate together to greatly improve both system performance and fairness. Our evaluations show that SMS not only reduces inter-application interference caused by the GPU, thereby improving heterogeneous system performance, but also provides better scalability and power efficiency compared to multiple state-of-the-art memory schedulers.

Third, we redesign the GPU memory management unit to efficiently handle new problems caused by the massive address translation parallelism present in GPU computation units in multi-GPU-application environments. Running multiple GPGPU applications concurrently induces significant inter-core thrashing on the shared TLB, a new phenomenon that we call inter-address-space interference. To reduce this interference, we introduce Multi Address Space Concurrent Kernels (MASK). MASK introduces TLB-awareness throughout the GPU memory hierarchy and introduces TLB- and cache-bypassing techniques to increase the effectiveness of a shared TLB.

Finally, we introduce MOSAIC, a hardware-software cooperative technique that further increases the effectiveness of TLB by modifying the memory allocation policy in the system software. MOSAIC introduces a high-throughput method to support large pages in multi-GPU-application environments. Our evaluations show that the MASK-MOSAIC combination provides a simple mechanism that eliminates the performance overhead of address translation in GPUs without significant changes to GPU hardware, thereby greatly improving GPU system performance.

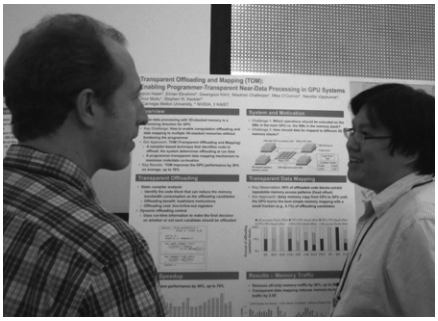
The key conclusion of this dissertation is that a combination of GPU-aware cache and memory management techniques can effectively mitigate the memory interference on current and future GPU-based systems.

DISSERTATION ABSTRACT: Exploiting Application Characteristics for Efficient System Support of Data-Parallel Machine Learning

Henggang Cui
Carnegie Mellon University, ECE

PhD Defense — April 6, 2017

Large scale machine learning has many characteristics that can be exploited in the system designs to improve its efficiency. This dissertation demonstrates that, the characteristics of the ML computations can be exploited in the design and implementation of parameter server systems, to greatly improve the efficiency by an order of magnitude or more. We support this thesis statement with three case study systems, IterStore, GeePS, and MLtuner. IterStore is an optimized parameter server system design that exploits the repeated data access pattern characteristic of ML computations. The designed optimizations allow IterStore to reduce the total run time of our ML benchmarks by up to 50X. GeePS is a parameter server that is specialized for deep learning on distributed GPUs. By exploiting the layer-by-layer data access and computation pattern of deep learning, GeePS provides almost linear scalability from single-machine baselines (13 more training throughput with 16 machines), and also supports neural networks that do not fit in GPU memory. MLtuner is a system for automatically tuning the training tunables of ML tasks. It exploits the characteristic that the best tunable settings can often be decided quickly with just a short trial time. By making use of optimization-guided online trial-and-error, MLtuner can robustly find and re-tune tunable settings for a variety of machine learning applications, including image classification, video classification, and matrix factorization, and is over an order of magnitude faster than traditional hyper-parameter tuning approaches.



Kevin Hsieh tells Michael Kozuch of Intel about his research on "Gaia: Geo-Distributed Machine Learning Approaching LAN Speeds" at a retreat poster session.

DISSERTATION ABSTRACT: Scaling Distributed Machine Learning with System and Algorithm Co-design

Mu Li
Carnegie Mellon University, ML

PhD Defense — February 2, 2017

Due to the rapid growth of data and the ever increasing model complexity, which often manifests itself in the large number of model parameters, today, many important machine learning problems cannot be efficiently solved by a single machine. Distributed optimization and inference is becoming more and more inevitable for solving large scale machine learning problems in both academia and industry. However, obtaining an efficient distributed implementation of an algorithm, is far from trivial. Both intensive computational workloads and the volume of data communication demand careful design of distributed computation systems and distributed machine learning algorithms. In this thesis, we focus on the co-design of distributed computing systems and distributed optimization algorithms that are specialized for large machine learning problems.

In the first part, we propose two distributed computing frameworks: Parameter Server, a distributed machine learning framework that features efficient data

continued on page 16

continued from page 15

communication between the machines; MXNet, a multi-language library that aims to simplify the development of deep neural network algorithms. We have witnessed the wide adoption of the two proposed systems in the past two years. They have enabled and will continue to enable more people to harness the power of distributed computing to design efficient large-scale machine learning applications.

In the second part, we examine a number of distributed optimization problems in machine learning, leveraging the two computing platforms. We present new methods to accelerate the training process, such as data partitioning with better locality properties, communication friendly optimization methods, and more compact statistical models. We implement the new algorithms on the two systems and test on large scale real data sets. We successfully demonstrate that careful co-design of computing systems and learning algorithms can greatly accelerate large scale distributed machine learning.

DISSERTATION ABSTRACT: Prescriptive Safety-Checks through Automated Proofs for Control-Flow Integrity

Jiaqi Tan
Carnegie Mellon University, ECE

PhD Defense — November 2016

Embedded software today is pervasive: they can be found everywhere, from coffee makers and medical devices, to cars and aircraft. Embedded software today is also open and connected to the Internet, exposing them to external attacks that can cause its Control-Flow Integrity (CFI) to be violated. Control-Flow Integrity is an important safety property of software, which ensures that the behavior of the software is not inadvertently changed. The violation of CFI in software can cause unintended behaviors, and can even lead to catastrophic incidents in safety-critical systems.

This dissertation develops a two-part approach for CFI: (i) prescribing source-code safety checks, that prevent the root-causes of CFI, that programmers can insert themselves, and (ii) formally proving CFI for the machine-code of programs with source-code safety-checks. First, our prescribed safety-checks, when applied, prevent the root-causes of CFI, thereby enabling software to recover from CFI violations in a customizable way. In addition, our prescribed safety-checks are visible to programmers, empowering them to ensure that the behavior of their software is not inadvertently changed by the prescribed safety-checks. However, programmer-inserted safety-checks may be incomplete. Thus, current techniques for proving CFI, which assume that safety-checks are complete, may not work. Second, this dissertation develops a logic approach that automates formal proofs of CFI for the machine-code of software containing both source-code CFI safety-checks and system calls. We extend an existing trustworthy Hoare logic with new proof rules, proof tactics, and a novel proof-search algorithm, which exploit the principle of local reasoning for safety properties to automatically generate CFI proofs for the machine-code of programs compiled with our prescribed source-code safety-checks.

To the best of our knowledge, our approach to CFI is the first to combine programmer-visible source-code enforcement mechanisms for CFI–enabling programmers to customize them and observe that their software is not inadvertently changed–with machine-code proofs of CFI that can be automated, and that does not require a trusted or verified compiler to ensure its proven properties hold in machine-code.

Abutalib Aghayev presents his research on "Evolving ext4 for SMR Disks" at the 2016 PDL Retreat.

We evaluate our CFI approach on realistic embedded software. We evaluate our approach on the MiBench and WCET benchmarks, implementations of common file utilities, and programs interfacing with hardware inputs and outputs on the Raspberry Pi single-board-computer. The variety of our target programs, and our ability to support useful features such as file and hardware inputs and outputs, demonstrate the wide applicability of our approach.

DISSERTATION ABSTRACT: System Infrastructure for Mobile-Cloud Convergence

Kiryong Ha
Carnegie Mellon University, SCS

PhD Defense — October 20, 2016

The convergence of mobile computing and cloud computing enables new mobile applications that are both resource-intensive and interactive. For these applications, end-to-end network bandwidth and latency matter greatly when cloud resources are used to augment the computational power and battery life of a mobile device. This dissertation designs and implements a new architectural element called a cloudlet, that arises from the convergence of mobile computing and cloud computing. Cloudlets represent the middle tier of a 3-tier hierarchy, mobile device - cloudlet - cloud, to achieve the right balance between cloud consolidation and network responsiveness. We first present quantitative evidence that shows cloud location can affect the performance of mobile applications and cloud consolidation.

We then describe an architectural solution using cloudlets that are a seamless

continued on page 17

continued from page 16

extension of today's cloud computing infrastructure. Finally, we define minimal functionalities that cloudlets must offer above/beyond standard cloud computing, and address corresponding technical challenges.

DISSERTATION ABSTRACT: Scheduling with Space-Time Soft Constraints in Heterogeneous Cloud Datacenters

Alexey Tumanov
Carnegie Mellon University, ECE

PhD Defense — July 25, 2016

Heterogeneity in modern datacenters is on the rise, in hardware resource characteristics, in workload characteristics, and in dynamic characteristics (e.g., a memory-resident copy of input data). As a result, which machines are assigned to a given job can have a significant impact. For example, a job may run faster on the same machine as its input data or with a given hardware accelerator, while still being runnable on other machines, albeit less efficiently. Heterogeneity takes on more complex forms as sets of resources differ in the level of performance they deliver, even if they consist of identical individual units, such as with rack-level locality. We refer to this as combinatorial heterogeneity. Mixes of jobs with strict SLOs on completion time and increasingly available runtime estimates in production datacenters deepen the challenge of matching the right resources to the right workloads at the right time.

In this dissertation, we hypothesize that it is possible and beneficial to simultaneously leverage all of this information in the form of declaratively specified spacetime soft constraints. To accomplish this, we first design and develop our principal building block—a novel Space-Time Request Language (STRL). It enables the expression of jobs' preferences and flexibility in a general, extensible way by using a declarative, composable, intuitive algebraic expression structure. Second, building on the generality of STRL, we propose an equally general STRL Compiler that automatically compiles STRL expressions into Mixed Integer Linear Programming (MILP) problems that can be aggregated and solved to maximize the overall value of shared cluster resources.

These theoretical contributions form the foundation for the system we architect, called TetriSched, that instantiates our conceptual contributions: (a) declarative soft constraints, (b) space-time soft constraints, (c) combinatorial constraints, (d) orderless global scheduling, and (e) in situ preemption. We also propose a set of mechanisms that extend the scope and the practicality of TetriSched's deployment by analyzing and improving on its scalability, enabling and studying the efficacy of preemption, and featuring a set of runtime misestimation handling mechanisms to address runtime prediction inaccuracy.

In collaboration with Microsoft, we adapt some of these ideas as we design and implement a heterogeneity-aware resource reservation system called Aramid with support for ordinal placement preferences targeting deployment in production clusters at Microsoft scale. A combination of simulation and real cluster experiments with synthetic and production-derived workloads, a range of workload intensities, degrees of burstiness, preference strengths, and input inaccuracies support our hypothesis that leveraging space-time soft constraints (a) significantly improves scheduling quality and (b) is possible to achieve in a practical deployment.

DISSERTATION ABSTRACT: Practical Data Compression for Modern Memory Hierarchies

Gennady G. Pekhimenko
Carnegie Mellon University, SCS

PhD Defense — July 1, 2016

Although compression has been widely used for decades to reduce file sizes (conserving storage capacity and network bandwidth when moving files), there has been little to no use within modern memory hierarchies. Why not? As programs become increasingly data-intensive, the capacity and bandwidth within the memory hierarchy (including caches, main memory, and their associated interconnects) are becoming increasingly important bottlenecks. If data compression were applied successfully to the memory hierarchy, it could relieve pressure on these bottlenecks by increasing effective capacity, increasing effective bandwidth, even reducing energy consumption.

In this thesis, I describe a new, practical approach to integrating data compression within the memory hierarchy, including on-chip caches, main memory, and both on-chip and off-chip interconnects. This new approach is fast, simple, and effective in saving storage space. A key insight in our approach is that access time (including decompression latency) is critical in modern memory hierarchies. By combining inexpensive hardware support with modest OS support, our holistic approach to compression achieves substantial improvements in performance and energy efficiency across the memory hierarchy. In addition to exploring compression-related issues and enabling practical solutions in modern CPU systems, we discover new problems in realizing hardware-based compression for GPU-based systems and develop new solutions to solve these problems.

continued on page 18

continued from page 17

THESIS PROPOSAL: Practical Concurrency Testing

Benjamin Blum, SCS
April 25, 2017

Concurrent programming presents a challenge to students and experts alike because of the complexity of multi-threaded interactions and the difficulty to reproduce and reason about bugs. Stateless model checking is a concurrency testing approach which forces a program to interleave its threads in many different ways, checking for bugs each time. This technique is powerful, in principle capable of finding any nondeterministic bug in finite time, but suffers from exponential explosion as program size increases. Checking an exponential number of thread interleavings is not a practical or predictable approach for programmers to find concurrency bugs before their project deadlines.

In this thesis, I propose to make stateless model checking more practical for human use by way of several new techniques. I have built Landslide, a stateless model checker specializing in student projects for undergraduate operating systems classes. Landslide includes a novel algorithm for automatically managing multiple state spaces according to their estimated completion times, which I will show quickly finds bugs should they exist and also quickly verifies correctness otherwise. I will evaluate Landslide's suitability for inexpert use by presenting the results of many semesters providing it to students in 15-410, CMU's Operating System Design and Implementation class. Finally, I will explore broader impact by extending Landslide to test some real-world programs and to be used by students at other universities.

THESIS PROPOSAL: Scaling Distributed Machine Learning with System and Algorithm Co-design

Mu Li, SCS
October 14, 2016

For a lot of important machine learning problems, due to the rapid growth of data and the ever increasing model complexity, which often manifests itself in the large number of model parameters, no single machine can solve them fast enough. Thus, distributed optimization and inference is becoming more inevitable for solving large scale machine learning problems in both academia and industry. Obtaining an efficient distributed implementation of an algorithm, however, is far from trivial. Both intensive computational workloads and the volume of data communication demand careful design of distributed computation systems and distributed machine learning algorithms.

In this thesis, we focus on the co-design of distributed computing systems and distributed optimization algorithms that are specialized for large machine learning problems. We propose two distributed computing frameworks: a parameter server framework which features efficient data communication, and MXNet, a multi-language library aiming to simplify the development of deep neural network algorithms. In less than two years, we have witnessed the wide adoption of the proposed systems. We believe that as we continue to develop these systems, they will enable more people to take advantage of the power of distributed computing to design efficient machine learning applications to solve large-scale computational problems. Leveraging the two computing platforms, we examine a number of distributed optimization problems in machine learning. We present new methods to accelerate the training process, such as data partitioning with better locality properties, communication friendly optimization methods, and more compact statistical models. We implement the new algorithms on the two systems and test on large scale real data sets. We successfully demonstrate that careful co-design of computing systems and learning algorithms can greatly accelerate large scale distributed machine learning.

THESIS PROPOSAL: Architectural Techniques for Improving NAND Flash Memory Reliability

Yixin Luo, SCS
August 5, 2016

Raw bit errors are common in NAND flash memory and will increase in the future. These errors reduce flash reliability and limit the lifetime of a flash memory device. This proposal aims to improve flash reliability with a multitude of low-cost architectural techniques. Our thesis statement is: NAND flash memory reliability can be improved at low cost and with low performance overhead by deploying various architectural techniques that are aware of higher-level application behavior and underlying flash device characteristics.

Our proposed approach is to understand flash error characteristics and workload behavior through characterization, and to design smart flash controller algorithms that utilize this understanding to improve flash reliability. We propose to investigate four directions through this approach. (1) Our preliminary work proposes a new technique that improves flash reliability by 12.9 times by managing flash retention differently for write-hot data and write-cold data. (2) We propose to characterize and model flash errors on new flash chips. (3) We propose to develop a technique to construct a flash error model online and improve flash lifetime by exploiting our online model. (4) We propose to understand and develop new techniques that utilize flash self-healing effect. We hope that these four directions will allow us to achieve higher flash reliability at low cost.

Nathan Beckmann

The PDL would like to welcome Nathan Beckmann to our family! Nathan is an Assistant Professor at CMU in the Department of Computer Science. Nathan works in computer architecture, with experience in distributed operating systems and is currently focused on reducing data movement in multicores, which uses over half of the energy available in current chips, by redesigning caches so they dynamically adopt an application-specific organization.

Nathan received his PhD from MIT's Computer Science and Artificial Intelligence Laboratory (CSAIL)(2012-2015), advised by Daniel Sanchez. His PhD thesis, "Design and Analysis of Spatially-Partitioned Shared Caches," received the 2015 Sprowls Doctoral Thesis Prize for the "best PhD thesis in computer science at MIT". This work exposes physical cache banks to software, letting the operating system schedule applications' working sets in nearby cache banks. This work spans cache partitioning, dynamic NUCA, heterogeneous memories, replacement policies, etc to reduce overall data movement. A common theme is the use of analytical models in software to control simple, efficient, but "dumb" hardware. This analytical approach yields robust performance and unexpected insights. Following graduation, he stayed for a one-year postdoc.

Before his PhD, Nathan worked in Anant Agarwal's CSAIL group on fos, a distributed operating system for multicore and clouds (2008-2011). He also briefly worked with Frans Kaashoek and Nickolai Zeldovich in the Parallel & Distributed Operating Systems Group (2011-2012). A long time ago, he also worked on Graphite, a distributed multicore simulator. Other interests include math and cryptography.

Nathan received his masters degree in 2010 from MIT. His thesis, "Distributed Naming in a Factored Operating System", won the William A. Martin Memorial Thesis Award for an outstanding Master's thesis. He received his Bachelor of Science from the University of California, Los Angeles, graduating summa cum laude in Computer Science and Mathematics of Computation and was honored as the Bachelor of the Year (2008) in Computer Science.

ALUMNI NEWS

Vinaykumar Bhat (MS, INI '15)

PDL Alumni Vinaykumar Bhat (2014-15) and Preeti Murthy (also a CMU ECE graduate), who both took the much-in-demand PDL Storage Systems class, were married on March 16th. Vinay currently works for Oracle's in-memory database group.

Peter Klemperer (PhD, ECE '14)

Starting this coming Fall, Peter will be entering a new role as an Assistant Professor of Computer Science at Mount Holyoke College in South Hadley, MA. He will be leading a 3-year pilot into whether Engineering has a role at Mount Holyoke. There is no Engineering Department at Mount Holyoke yet, but many students enroll in dual-degree programs with nearby schools that do offer one. His role will be to provide Engineering experiences at Mount Holyoke that help develop interest, facilitate transition into the dual-degree programs, and help integrate their skills back into the Mount Holyoke College community. He also got a new dog. Her name is Hope.

Atreyee Maiti (MS, SCS '14)

Atreyee, now a Senior MTS with Nutanix, recently had a paper accepted by the International Conference on Cloud Engineering (IC2E) 2017 in the industrial track. The paper "Quest: Search-driven Management of Cloud-Scale Data Centers" was presented in April in Vancouver, B.C.

Niraj Tolia (PhD, SCS '07)

Niraj, who currently resides in the San Francisco Bay area, has recently founded a new startup (Kasten, kasten.io), working on container-based/cloud-native storage infrastructure. While they are still in stealth mode, they are hiring and have other PDL alumni on board too!

Amber Palekar (MS, INI '06)

Amber has moved on from his position as a file system engineer at NetApp. He is now working at Nutanix, moving over as a part of Nutanix's acquisition of PernixData. He says he is having some startup fun at the moment!

continued from page 7

to extract the data stored within the device. Instead, investigators turn to chip-off analysis, where they use a thermal-based procedure to physically remove the NAND flash memory chip from the device, and access the chip directly to extract the raw data stored on the chip.

We perform an analysis of the errors introduced into multi-level cell (MLC) NAND flash memory chips after the device has been seized. We make two major observations. First, between the time that a device is seized and the time digital forensic investigators perform data extraction, a large number of errors can be introduced as a result of charge leakage from the cells of the NAND flash memory (known as data retention errors). Second, when thermal-based chip removal is performed, the number of errors in the data stored within NAND flash memory can increase by two or more orders of magnitude, as the high temperature applied to the chip greatly accelerates charge leakage. We demonstrate that the chip-off analysis based forensic data recovery procedure is quite destructive, and can often render most of the data within NAND flash memory uncorrectable, and, thus, unrecoverable.

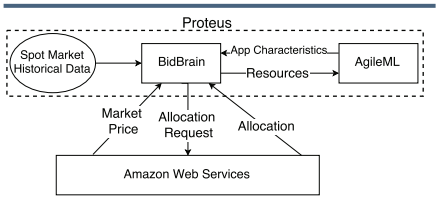
To mitigate the errors introduced during the forensic recovery process, we explore a new hardware-based approach. We exploit a fine-grained read reference voltage control mechanism implemented in modern NAND flash memory chips, called read-retry, which can compensate for the charge leakage that occurs due to (1) retention loss and (2) thermal-based chip removal. The read-retry mechanism successfully reduces the number of errors, such that the original data can be fully recovered in our tested chips as long as the chips were not heavily used prior to seizure. We conclude that the read-retry mechanism should be adopted as part of the forensic data recovery process.

Proteus: Agile ML Elasticity through Tiered Reliability in Dynamic Resource Markets

Aaron Harlap, Alexey Tumanov, Andrew Chung, Greg Ganger & Phil Gibbons

ACM European Conference on Computer Systems, 2017 (EuroSys'17), 23rd-26th April, 2017, Belgrade, Serbia. Supersedes Carnegie Mellon University Parallel Data Lab Technical Report CMU-PDL-16-102. May 2016.

Many shared computing clusters allow users to utilize excess idle resources at lower cost or priority, with the proviso that some or all may be taken away at any time. But, exploiting such dynamic resource availability and the often fluctuating markets for them requires agile elasticity and effective acquisition strategies. Proteus aggressively exploits such transient revocable resources to do machine learning (ML) cheaper and/or faster. Its parameter server framework, AgileML, efficiently adapts to bulk additions and revocations of transient machines, through a novel 3-stage active-backup approach, with minimal use of more costly non-transient resources. Its BidBrain component adaptively allocates resources from multiple EC2 spot markets to minimize average cost per work as transient resource availability and cost change over time. Our evaluations show that Proteus reduces cost by 85% relative to non-transient pricing, and by 43% relative to previous approaches, while simultaneously reducing runtimes by up to 37%.



The Proteus architecture consists of the resource allocation component, BidBrain, and the elastic ML framework, AgileML.

An Empirical Evaluation of In-Memory Multi-Version Concurrency Control

Yingjun Wu, Joy Arulraj, Jiexi Lin, Ran Xian & Andrew Pavlo

Proceedings of the VLDB Endowment, vol. 10, iss. 7, pages. 781—792, March 2017.

Multi-version concurrency control (MVCC) is currently the most popular transaction management scheme in modern database management systems (DBMSs). Although MVCC was discovered in the late 1970s, it is used in almost every major relational DBMS released in the last decade. Maintaining multiple versions of data potentially increases parallelism without sacrificing serializability when processing transactions. But scaling MVCC in a multi-core and in-memory setting is non-trivial: when there are a large number of threads running in parallel, the synchronization overhead can outweigh the benefits of multi-versioning.

To understand how MVCC perform when processing transactions in modern hardware settings, we conduct an extensive study of the scheme's four key design decisions: concurrency control protocol, version storage, garbage collection, and index management. We implemented state-of-the-art variants of all of these in an in-memory DBMS and evaluated them using OLTP workloads. Our analysis identifies the fundamental bottlenecks of each design choice.

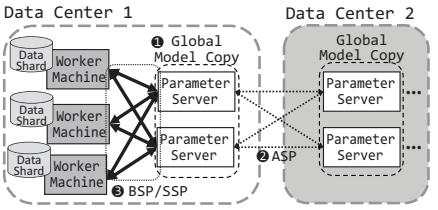
Gaia: Geo-Distributed Machine Learning Approaching LAN Speeds

Kevin Hsieh, Aaron Harlap, Nandita Vijaykumar, Dimitris Konomis, Gregory R. Ganger, Phillip B. Gibbons & Onur Mutlu

14th USENIX Symposium on Networked Systems Design and Implementation (NSDI), March 27–29, 2017, Boston, MA.

continued on page 21

continued from page 20



Gaia system overview.

Machine learning (ML) is widely used to derive useful information from large-scale data (such as user activities, pictures, and videos) generated at increasingly rapid rates, all over the world. Unfortunately, it is infeasible to move all this globally-generated data to a centralized data center before running an ML algorithm over it -- moving large amounts of raw data over a wide-area network (WAN) can be extremely slow, and is also subject to national privacy law constraints. This motivates the need for a geo-distributed ML system spanning multiple data centers. Unfortunately, communicating over WANs can significantly degrade ML system performance (by as much as 53.7X in our study) because the communication overwhelms the limited WAN bandwidth.

Our goal in this work is to develop a geo-distributed ML system that (1) employs an intelligent communication mechanism over WANs to efficiently utilize the scarce WAN bandwidth, while retaining the accuracy and correctness guarantees of an ML algorithm; and (2) is generic and flexible enough to run a wide range of ML algorithms, without requiring any changes to the algorithms.

To this end, we introduce a new, general geo-distributed ML system, Gaia, that decouples the communication within a data center from the communication between data centers, enabling different communication and consistency models for each. We present a new ML synchronization model, Approximate Synchronous Parallel (ASP), which dynamically adjusts to the available WAN bandwidth between data centers and eliminates the vast majority of insignificant communication while still guaranteeing the correctness of ML algorithms. Our experiments on our prototypes of Gaia running across the 11 Amazon EC2 global regions and on a cluster that emulates EC2 WAN bandwidth show that Gaia provides 1.8-53.5X speed-up over a state-of-art distributed ML system, and is within 0.94-1.56X of ML-on-LAN speeds.

Towards Edge-caching for Image Recognition

Utsav Drolia, Katherine Guo, Jiaqi Tan, Rajeev Gandhi & Priya Narasimhan

First Workshop on Smart Edge Computing and Networking (SmartEdge) '17, held in conjunction with PerCom 2017, March 13 - 17, 2017, Hawaii, USA.

With the available sensors on mobile devices and their improved CPU and storage capability, users expect their devices to recognize the surrounding environment and to provide relevant information and/or content automatically and immediately. For such classes of real-time applications, user perception of performance is key. To enable a truly seamless experience for the user, responses to requests need to be provided with minimal user-perceived latency.

Current state-of-the-art systems for these applications require offloading requests and data to the cloud. This paper proposes an approach to allow users' devices and their onboard applications to leverage resources closer to home, i.e., resources at the edge of the network. We propose to use edge-servers as specialized caches for image-recognition applications. We develop a detailed formula for the expected latency for such a cache that incorporates the effects of recognition algorithms' computation time and accuracy. We show that, counter-intuitively, large cache sizes can lead to higher latencies. To the best of our knowledge, this is the first work that models edge-servers as caches for compute-intensive recognition applications.

SoftMC: A Flexible and Practical Open-Source Infrastructure for Enabling Experimental DRAM Studies

Hasan Hassan, Nandita Vijaykumar, Samira Khan, Saugata Ghose, Kevin Chang, Gennady Pekhimenko, Donghyuk Lee, Oguz Ergin & Onur Mutlu

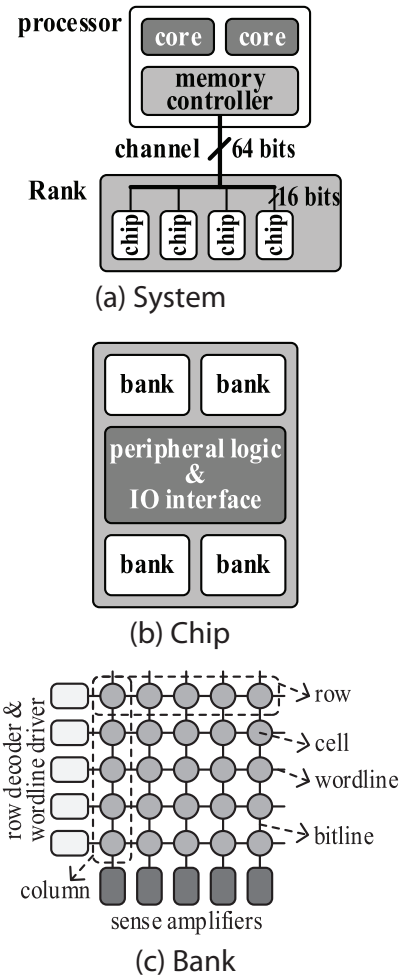
International Symposium on High-Performance Computer Architecture (HPCA), February 2017.

DRAM is the primary technology used for main memory in modern systems. Unfortunately, as DRAM scales down to smaller technology nodes, it faces key challenges in both data integrity and latency, which strongly affects overall system reliability and performance. To develop reliable and high-performance DRAM-based main memory in future systems, it is critical to characterize, understand, and analyze various aspects (e.g., reliability, latency) of existing DRAM chips. To enable this, there is a strong need for a publicly-available DRAM testing infrastructure that can flexibly and efficiently test DRAM chips in a manner accessible to both software and hardware developers.

This paper develops the first such infrastructure, SoftMC (Soft Memory Controller), an FPGA-based testing platform that can control and test memory modules designed for the commonly-used DDR (Double Data Rate) interface. SoftMC has two key properties: (i) it provides flexibility to thoroughly control memory behavior or to implement a wide range of mechanisms using DDR commands; and (ii) it is easy to use as it provides a simple and intuitive high-level programming interface for users, completely hiding the low-level details of the FPGA.

continued on page 22

continued from page 21



DRAM-based memory system organization.

We demonstrate the capability, flexibility, and programming ease of SoftMC with two example use cases. First, we implement a test that characterizes the retention time of DRAM cells. Experimental results we obtain using SoftMC are consistent with the findings of prior studies on retention time in modern DRAM, which serves as a validation of our infrastructure. Second, we validate two recently-proposed mechanisms, which rely on accessing recently-refreshed or recently-accessed DRAM cells faster than other DRAM cells. Using our infrastructure, we show that the expected latency reduction effect of these mechanisms is not observable in existing DRAM chips, which demonstrates the usefulness of SoftMC in testing new ideas on existing memory modules. We discuss several other use cases of SoftMC, including the ability to characterize emerging non-volatile memory modules that obey the DDR standard. We hope that our open-source release of SoftMC fills a gap in the space of publicly-available experimental memory testing infrastructures and inspires new studies, ideas, and methodologies in memory system design.

An Evaluation of Distributed Concurrency Control

Rachael Harding, Dana Van Aken, Andrew Pavlo & Michael Stonebraker

Proceedings of the VLDB Endowment, vol. 10, iss. 5, pages. 553—564, January 2017.

Increasing transaction volumes have led to a resurgence of interest in distributed transaction processing. In particular, partitioning data across several servers can improve throughput by allowing servers to process transactions in parallel. But executing transactions across servers limits the scalability and performance of these systems.

In this paper, we quantify the effects of distribution on concurrency control protocols in a distributed environment. We evaluate six classic and modern protocols in an in-memory distributed database evaluation framework called Deneva, providing an apples-to-apples comparison between each. Our results expose severe limitations of distributed transaction processing engines. Moreover, in our analysis, we identify several protocol-specific scalability bottlenecks. We conclude that to achieve truly scalable operation, distributed concurrency control solutions must seek a tighter coupling with either novel network hardware (in the local area) or applications (via data modeling and semantically-aware execution), or both.

Vulnerabilities in MLC NAND Flash Memory Programming: Experimental Analysis, Exploits, and Mitigation Techniques

Yu Cai, Saugata Ghose, Yixin Luo, Ken Mai, Onur Mutlu & Erich F. Haratsch

23rd IEEE Symposium on High Performance Computer Architecture, Industrial session, February 2017.

Modern NAND flash memory chips provide high density by storing two bits of data in each flash cell, called a multi-level cell (MLC). An MLC partitions the threshold voltage range of a flash cell into four voltage states. When a flash cell is programmed, a high voltage is applied to the cell. Due to parasitic capacitance coupling between flash cells that are physically close to each other, flash cell programming can lead to cell-to-cell program interference, which introduces errors into neighboring flash cells. In order to reduce the impact of cell-to-cell interference on the reliability of MLC NAND flash memory, flash manufacturers adopt a two-step programming method, which programs the MLC in two separate steps. First, the flash memory partially programs the least significant bit of the MLC to some intermediate threshold voltage. Second, it programs the most significant bit to bring the MLC up to its full voltage state.

In this paper, we demonstrate that two-step programming exposes new reliability and security vulnerabilities. We experimentally characterize the effects of two-step programming using contemporary 1X-nm (i.e., 15–19nm) flash memory chips. We find that a partially-programmed flash cell (i.e., a cell where the second programming step has not yet been performed) is much more vulnerable to cell-to-cell interference and read disturb than a fully-programmed cell. We show that it is possible to exploit these vulnerabilities on solid-state drives (SSDs)

continued on page 23

continued from page 22

to alter the partially-programmed data, causing (potentially malicious) data corruption. Building on our experimental observations, we propose several new mechanisms for MLC NAND flash memory that eliminate or mitigate data corruption in partially-programmed cells, thereby removing or reducing the extent of the vulnerabilities, and at the same time increasing flash memory lifetime by 16%.

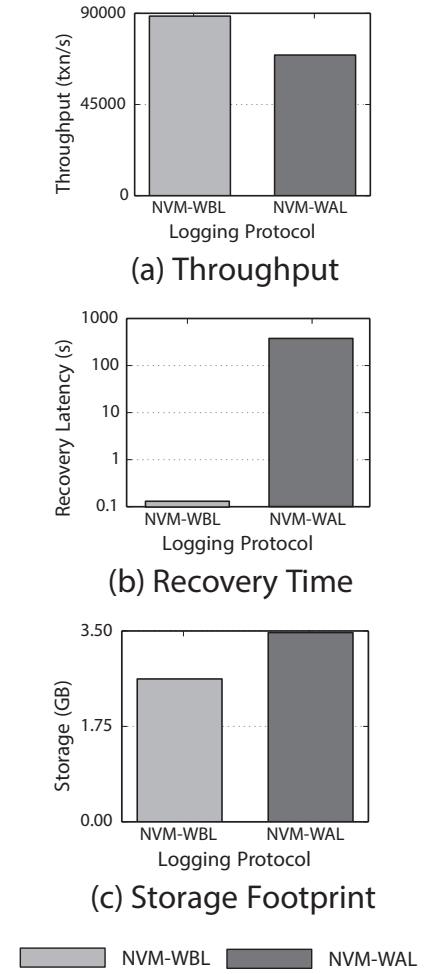
Write-Behind Logging

Joy Arulraj, Matthew Perron & Andrew Pavlo

Proc. VLDB Endow., vol. 10, pp. 337-348, December, 2016.

The design of the logging and recovery components of database management systems (DBMSs) has always been influenced by the difference in the performance characteristics of volatile (DRAM) and non-volatile storage devices (HDD/SSDs). The key assumption has been that non-volatile storage is much slower than DRAM and only supports block-oriented read/writes. But the arrival of new nonvolatile memory (NVM) storage that is almost as fast as DRAM with fine-grained read/writes invalidates these previous design choices.

This paper explores the changes that are required in a DBMS to leverage the unique properties of NVM in systems that still include volatile DRAM. We make the case for a new logging and recovery protocol, called write-behind logging, that enables a DBMS to recover nearly instantaneously from system failures. The key idea is that the DBMS logs what parts of the database have changed rather than how it was changed. Using this method, the DBMS flushes the changes to the database before recording them in the log. Our evaluation shows that this protocol improves a DBMS's transactional throughput by 1.3×, reduces the recovery time by more than two orders of magnitude, and shrinks the storage footprint of the DBMS on NVM by 1.5×. We also demonstrate that our logging protocol is compatible with standard replication schemes.



WBL vs. WAL – The throughput, recovery time, and storage footprint of the DBMS for the YCSB benchmark with the write-ahead logging and write-behind logging protocols.

FaSST: Fast, Scalable and Simple Distributed Transactions with Two-sided (RDMA) Datagram RPCs

Anuj Kalia, Michael Kaminsky & David G. Andersen

12th USENIX Symposium on Operating Systems Design and Implementation November 2–4, 2016, Savannah, GA, USA.

FaSST is an RDMA-based system that provides distributed in-memory transactions with serializability and durability. Existing RDMA-based transaction processing systems use one-sided RDMA primitives for their ability to bypass the remote CPU. This design choice brings several drawbacks. First, the limited flexibility of one-sided RDMA reduces performance and increases software complexity when designing distributed data stores. Second, deep-rooted technical limitations of RDMA hardware limit scalability in large clusters. FaSST eschews one-sided RDMA for fast RPCs using two-sided unreliable datagrams, which we show drop packets extremely rarely on modern RDMA networks. This approach provides better performance, scalability, and simplicity, without requiring expensive reliability mechanisms in software. In comparison with published numbers, FaSST outperforms FaRM on the TATP benchmark by almost 2x while using close to half the hardware resources, and it outperforms DrTM+R on the SmallBank benchmark by around 1.7x without making data locality assumptions.

Self-Driving Database Management Systems

A. Pavlo, G. Angulo, J. Arulraj, H. Lin, J. Lin, L. Ma, P. Menon, T. Mowry, M. Perron, I. Quah, S. Santurkar, A. Tomasic, S. Toor, D. V. Aken, Z. Wang, Y. Wu, R. Xian & T. Zhang

In CIDR 2017, Conference on Innovative Data Systems Research. January 8-11, 2017, Chaminade, CA.

In the last two decades, both researchers and vendors have built advisory tools to assist database administrators (DBAs) in various aspects of system tuning and physical design. Most of this previous work, however, is incomplete because they still require humans to make the final decisions about any changes to the database and are re-

continued on page 24

continued from page 23

actionary measures that fix problems after they occur.

What is needed for a truly "self-driving" database management system (DBMS) is a new architecture that is designed for autonomous operation. This is different than earlier attempts because all aspects of the system are controlled by an integrated planning component that not only optimizes the system for the current workload, but also predicts future workload trends so that the system can prepare itself accordingly. With this, the DBMS can support all of the previous tuning techniques without requiring a human to determine the right way and proper time to deploy them. It also enables new optimizations that are important for modern high-performance DBMSs, but which are not possible today because the complexity of managing these systems has surpassed the abilities of human experts.

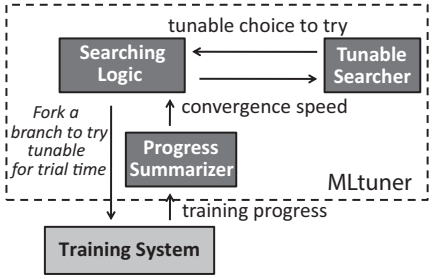
This paper presents the architecture of Peloton, the first self-driving DBMS. Peloton's autonomic capabilities are now possible due to algorithmic advancements in deep learning, as well as improvements in hardware and adaptive database architectures.

MLtuner: System Support for Automatic Machine Learning Tuning

Henggang Cui, Gregory R. Ganger & Phillip B. Gibbons

Carnegie Mellon University Parallel Data Lab Technical Report CMU-PDL-16-108, October 2016.

MLtuner automatically tunes settings for training tunables—such as the learning rate, the mini-batch size, and the data staleness bound—that have a significant impact on large-scale machine learning (ML) performance. Traditionally, these tunables are set manually, which is unsurprisingly error prone and difficult to do without extensive domain knowledge. MLtuner uses efficient snapshotting and optimization-guided online trial-and-error to find good initial settings as well as to re-tune settings during execution. Experiments with three real ML tasks show that MLtuner automatically enables performance within 40–178% of having oracle knowledge of the best settings, and outperforms oracle when no single set of settings are best for the entire execution. It also significantly outperforms most of the many feasible settings that might get used in practice.



Tunable searching procedure.

Benchmarking Apache Spark with Machine Learning Applications

Jinliang Wei, Jin Kyu Kim & Garth A. Gibson

Carnegie Mellon University Parallel Data Lab Technical Report CMU-PDL-16-107 October 2016.

We benchmarked Apache Spark with a popular parallel machine learning training application, Distributed Stochastic Gradient Descent for Matrix Factorization [5] and compared the Spark implementation with alternative approaches for communicating model parameters, such as scheduled pipelining using POSIX socket or MPI, and distributed shared memory (e.g. parameter server [13]). We found that Spark performance suffers substantial overhead with only modest model size (rank of a few hundreds). For example, the PySpark implementation using one single-core executor was about 3X slower than a serial out-of-core Python implementation and 226X slower than a serial C++ implementation. With a modest dataset (Netflix dataset containing 100 million ratings), the PySpark implementation showed 5.5X speedup from 1 to 8 machines, using 1 core per machine. But it failed to achieve further speedup with more machines or gain speedup from using more cores per machine. While it's still ongoing investigation, we believed that shuffling as Spark's only scalable mechanism of propagating model updates is responsible for much of the overhead and more efficient approaches for communication could lead to much better performance.

Zorua: A Holistic Approach to Resource Virtualization in GPUs

Nandita Vijaykumar, Kevin Hsieh, Gennady Pekhimenko, Samira Khan, Saugata Ghose, Ashish Shrestha, Adwait Jog, Phillip B. Gibbons & Onur Mutlu

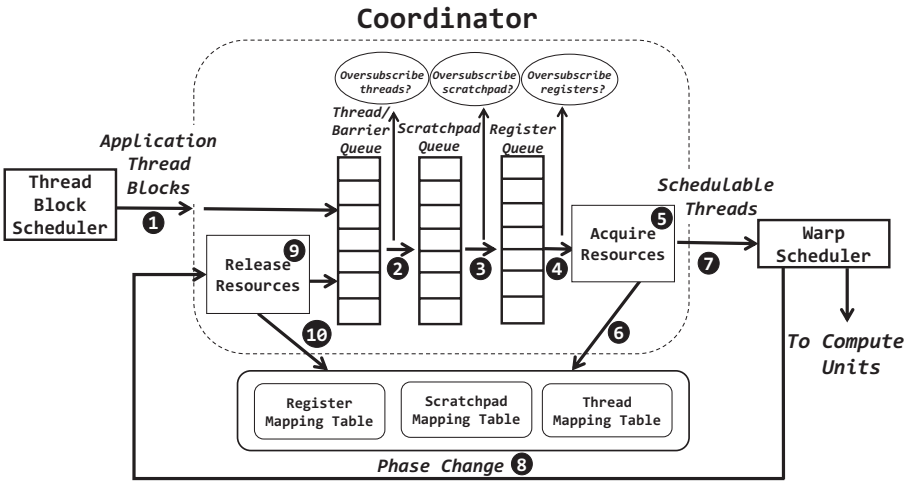
49th IEEE/ACM International Symposium on Microarchitecture (MICRO'16), October 15 - 19, 2016, Taipei, Taiwan.

This paper introduces a new resource virtualization framework, Zorua, that decouples the programmer-specified resource usage of a GPU application from the actual allocation in the on-chip hardware resources. Zorua enables this decoupling by virtualizing each resource transparently to the programmer. The virtualization provided by Zorua builds on two key concepts—dynamic allocation of the on-chip resources and their oversubscription using a swap space in memory.

Zorua provides a holistic GPU resource virtualization strategy, designed to (i) adaptively control the extent of oversubscription, and (ii) coordinate the dynamic management of multiple on-chip resources (i.e., registers, scratchpad memory, and thread slots), to maximize the effectiveness of virtu-

continued on page 25

continued from page 24



Overview of Zorua in hardware.

alization. Zorua employs a hardware-software codesign, comprising the compiler, a runtime system and hardware-based virtualization support. The runtime system leverages information from the compiler regarding resource requirements of each program phase to (i) dynamically allocate/deallocate the different resources in the physically available on-chip resources or their swap space, and (ii) manage the tradeoff between higher thread-level parallelism due to virtualization versus the latency and capacity overheads of swap space usage.

We demonstrate that by providing the illusion of more resources than physically available via controlled and coordinated virtualization, Zorua offers several important benefits: (i) Programming Ease. Zorua eases the burden on the programmer to provide code that is tuned to efficiently utilize the physically available on-chip resources. (ii) Portability. Zorua alleviates the necessity of re-tuning an application's resource usage when porting the application across GPU generations. (iii) Performance. By dynamically allocating resources and carefully oversubscribing them when necessary, Zorua improves or retains the performance of applications that are already highly tuned to best utilize the hardware resources. The holistic virtualization provided by Zorua can also enable other uses, including fine-grained resource sharing among multiple kernels and low-latency preemption of GPU programs.

Aging Gracefully with Geriatrix: A File System Aging Suite

Saurabh Kadekodi, Vaishnavh Nagarajan & Garth A. Gibson

Carnegie Mellon University Parallel Data Lab Technical Report CMU-PDL-16-105. October, 2016.

File system aging has been advocated for thorough analysis of any design, but it is cumbersome and often bypassed. Our aging study re-evaluates published file systems after aging using the same benchmarks originally used. We see significant performance degradation on HDDs and SSDs. With performance of aged file systems on SSDs dropping by as much as 80% relative to the recreated results of prior papers, aging is even more necessary in the era of SSDs. Still more concerning, the rank ordering of compared file systems can change versus published results.

We offer Geriatrix, a simple-to-use aging suite with built-in aging profiles with the goal of making it easier to age, and harder to justify ignoring file system aging in storage research.

A Survey of Security Vulnerabilities in Bluetooth Low Energy Beacons

Hui Jun Tay, Jiaqi Tan & Priya Narasimhan

Carnegie Mellon University Parallel Data Lab Technical Report CMU-PDL-16-109. November 2016.

There are currently 4 million Bluetooth Low Energy beacons, such as Apple's iBeacons™, deployed worldwide, with more being used in an increasingly wide variety of fields, from marketing to navigation. Yet, there is a lack of focus on the impact of security on beacon deployments. This report looks to capture an overview of current beacon deployments, the current vulnerabilities and risks present in using such platforms, and the security measures taken by vendors today.

A Model for Application Slowdown Estimation in On-Chip Networks and Its Use for Improving System Fairness and Performance

Xiyue Xiang, Saugata Ghose, Onur Mutlu & Nian-Feng Tzeng

International Conference on Computer Design (ICCD), October 3-5, 2016, Phoenix, USA.

In a network-on-chip (NoC) based system, the NoC is a shared resource among multiple processor cores. Network requests generated by different applications running on different cores can interfere with each other, leading to a slowdown in performance of each application. The degree of slowdown introduced by this interference varies for each application, as it depends on (1) the sensitivity of the application to

continued on page 26

continued from page 25

NoC performance, and (2) network traffic induced by other applications running concurrently on the system. In modern systems, NoC interference is largely uncontrolled, and therefore some applications unfairly slow down much more than others. This can lead to overall system performance degradation, prevent fair progress of different applications, and cause starvation of unfairly-treated applications. Our goal is to accurately model the slowdown of each application executing on the system due to NoC interference at runtime, and to use this information to improve system performance and reduce unfairness.

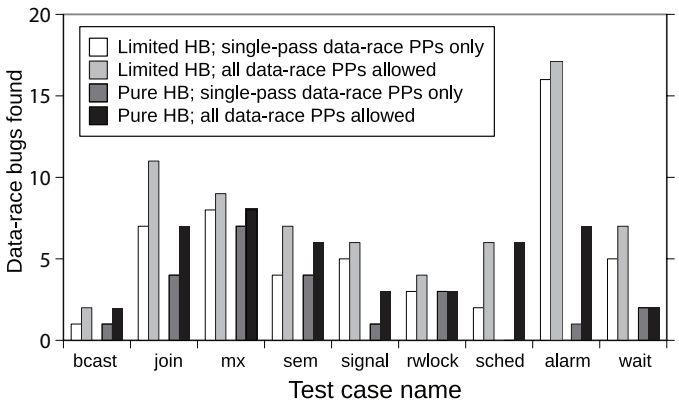
To this end, we propose the NoC Application Slowdown (NAS) Model, the first online model that accurately estimates how much network delays due to interference contribute to the overall stall time of each application. The key idea of NAS is to determine how the delays induced at each level of network data transmission overlap with each other, and to use the overlap information to calculate the net impact of the delays on application stall time. Our model determines the application slowdowns at runtime with a very low error rate, averaging 4.2% over 90 multiprogrammed workloads for an 88 mesh network. We use NAS to develop Fairness- Aware Source Throttling (FAST), a mechanism that employs slowdown predictions to control the network injection rates of applications in a way that minimizes system unfairness. Our results over a variety of multiprogrammed workloads show that FAST improves average system fairness and performance by 9.5% and 5.2%, respectively.

Stateless Model Checking with Data-Race Preemption Points

Ben Blum & Garth Gibson

SPLASH 2016 OOPSLA, Oct. 30 - Nov. 4, 2016, Amsterdam, Netherlands.

Stateless model checking is a powerful technique for testing concurrent programs, but suffers from exponential state space explosion when the test input parameters are too large. Several reduction techniques can mitigate this explosion, but even after pruning equivalent interleavings, the state space size is often intractable. Most prior tools are limited to preempting only on synchronization APIs, which reduces the space further, but can miss unsynchronized thread communication bugs. Data race detection, another concurrency testing approach, focuses on suspicious memory access pairs during a single test execution. It avoids concerns of state space size, but may report races that do not lead to observable failures, which jeopardizes a user's willingness to use the analysis.



Some data-race candidates may not be identified during a single program execution. Using nondeterministic data races as PPs, QUICKSAND found 128% (Limited HB) to 191% (Pure HB) as many data-race bugs compared to using single-pass candidates alone.

We present QUICKSAND, a new stateless model checking framework which manages the exploration of many state spaces using different preemption points. It uses state space estimation to prioritize jobs most likely to complete in a fixed CPU budget, and it incorporates data-race analysis to add new preemption points on the fly. Preempting threads during a data race's instructions can automatically classify the race as buggy or benign, and uncovers new bugs not reachable by prior model checkers. It also enables full verification of all possible schedules when every data race is verified as benign within the CPU budget. In our evaluation, QUICKSAND found 1.25x as many bugs and verified 4.3x as many tests compared to prior model checking approaches.

JamaisVu: Robust Scheduling with Auto-Estimated Job Runtimes

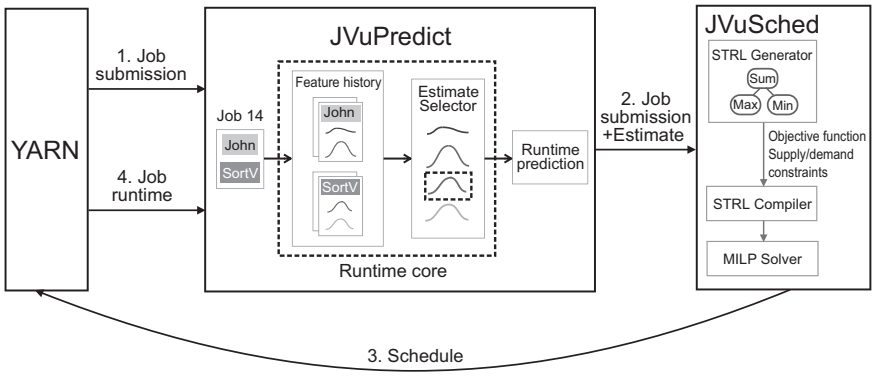
Alexey Tumanov, Angela Jiang, Jun Woo Park, Michael A. Kozuch & Gregory R. Ganger

Carnegie Mellon University Parallel Data Laboratory Technical Report CMU-PDL-16-104, September 2016.

JamaisVu is a new end-to-end cluster scheduling system that automatically generates and robustly exploits job runtime predictions. Using runtime knowledge allows it to more effectively pack jobs with diverse time concerns (e.g., deadline vs. latency) and soft-placement constraints on heterogeneous cluster resources. JamaisVu's job run time predictor, JVuPredict uses a new black-box approach that tracks job run time history as a function of multiple job submission features (e.g., user ID and program name), and then adaptively uses the most effective feature subset for each submitted job. Analysis of a 1-month Google cluster trace shows JVuPredict predicts reasonably well for complex real-world job mixes; for example, 90% of predictions are within a factor of two of actual runtime. But, because predictions cannot be perfect, JamaisVu includes new techniques for mitigating the effects of such real misprediction profiles. Experiments with

continued on page 27

continued from page 26



End-to-end system integration: JVuSched is integrated into Hadoop YARN—a popular open source cluster scheduling framework.

workloads derived from the trace show that JamaisVu performs nearly as well as a hypothetical scheduler with perfect job runtime information, outperforming runtime-unaware scheduling by reducing SLO miss rate, increasing goodput, and maintaining comparable latency for best effort jobs.

AUSPICE-R: Automatic Safety-Property Proofs for Realistic Features in Machine Code

Jiaqi Tan, Hui Jun Tay, Rajeev Gandhi, & Priya Narasimhan

14th Asian Symposium on Programming Languages and Systems (ASPLAS), November, 2016.

Verification of machine-code programs using program logic has focused on functional correctness, and proofs have required manually-provided program specifications. Fortunately, the verification of shallow safety properties such as memory and control-flow safety can be easier to automate, but past techniques for automatically verifying machine-code safety have required post-compilation transformations, which can change program behavior. In this work, we automatically verify safety properties for unmodified machine-code programs without requiring user-supplied specifications. We present our novel logic framework, AUSPICE, for automatic safety property verification for unmodified executables, which extends an existing trustworthy Hoare logic for local reasoning, and provides a novel proof tactic for selective composition. We demonstrate our fully automated proof technique on synthetic and realistic programs, and our verification completes in 6 hours for a realistic 533-instruction string search algorithm, demonstrating the feasibility of our approach.

PCFIRE: Towards Provable, Preventative Control-Flow Integrity Enforcement for Realistic Embedded Software

Jiaqi Tan, Hui Jun Tay, Utsav Drolia, Rajeev Gandhi & Priya Narasimhan.

ACM SIGBED International Conference on Embedded Software (EMSOFT), October 2016.

Control-Flow Integrity (CFI) is an important safety property of software, particularly in embedded and safety-critical systems, where CFI violations have led to patient deaths and can render cars remotely controllable by attackers. Previous techniques for CFI may reduce the robustness of embedded and safety-critical systems, as they handle CFI violations by stopping programs. In this work, we present PCFIRE, a preventative approach to CFI that prevents the root-causes of CFI violations to allow recovery, and enables programmers to specify robust recovery actions by providing CFI via source-code safety-checks. PCFIRE's CFI can be formally proved automatically, and supports realistic features of embedded software such as hardware and I/O access. We showcase PCFIRE by providing, and automatically proving, CFI for: benchmark programs, text utilities containing I/O, and embedded programs with sensor inputs and hardware outputs on the Raspberry Pi single-board computer.

μC-States: Fine-grained GPU Datapath Power Management

Onur Kayıran, Adwait Jog, Ashutosh Pattnaik, Rachata Ausavarungnirun, Xulong Tang, Mahmut T. Kandemir, Gabriel H. Loh, Onur Mutlu & Chita R. Das

Proceedings of the The 25th International Conference on Parallel Architectures and Compilation Techniques (PACT 2016), Haifa, Israel, September 2016.

To improve the performance of Graphics Processing Units (GPUs) beyond simply increasing core count, architects are recently adopting a scale-up approach: the peak throughput and individual capabilities of the GPU cores are increasing rapidly. This big-core trend in GPUs leads to various challenges, including higher static power consumption and lower and imbalanced utilization of the datapath components of a big core. As we show in this paper, two key problems ensue: (1) the lower and imbalanced datapath utilization can waste power as an application does not always utilize all portions of the big core datapath, and (2) the use of big cores can lead to application performance degradation in some cases due to the higher memory system contention caused by the more memory requests generated by each big core.

This paper introduces a new analysis of

continued on page 28

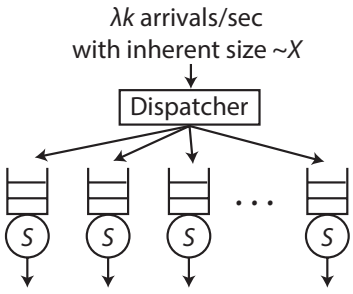
continued from page 27

datapath component utilization in big-core GPUs based on queuing theory principles. Building on this analysis, we introduce a fine-grained dynamic power- and clock-gating mechanism for the entire datapath, called µC-States, which aims to minimize power consumption by turning off or tuning-down datapath components that are not bottlenecks for the performance of the running application. Our experimental evaluation demonstrates that µC-States significantly reduces both static and dynamic power consumption in a big-core GPU, while also significantly improving the performance of applications affected by high memory system contention. We also show that our analysis of datapath component utilization can guide scheduling and design decisions in a GPU architecture that contains heterogeneous cores.

A Better Model for Job Redundancy: Decoupling Server Slowdown and Job Size

Kristen Gardner, Mor Harchol-Balter & Alan Scheller-Wolf

IEEE Modeling, Analysis and Simulation of Computer and Telecommunication Systems (MASCOTS 2016), London, UK, September 2016.



The S&X model. The system has k servers and jobs arrive as a Poisson process with rate λk. Each job has an inherent size X. When a job runs on a server it experiences slowdown S. A job's running time on a single server is R(1) = X·S. When a job runs on multiple servers, its inherent size X is the same on all these servers and it experiences a different, independently drawn instance of S on each server.

Recent computer systems research has proposed using redundant requests to reduce latency. The idea is to replicate a request so that it joins the queue at multiple servers. The request is considered complete as soon as any one copy of the request completes. Redundancy is beneficial because it allows us to overcome server-side variability – the fact that the server we choose might be temporarily slow due to factors such as background load, network interrupts, and garbage collection. When there is significant server-side variability, replicating requests can greatly reduce response times.

In the past few years, queueing theorists have begun to study redundancy, first via approximations, and, more recently, via exact analysis. Unfortunately, for analytical tractability, most existing theoretical analysis has assumed an Independent Runtimes (IR) model, wherein the replicas of a job each experience independent runtimes (service times) at different servers. The IR model is unrealistic and has led to theoretical results which can be at odds with computer systems implementation results. This paper introduces a much more realistic model of redundancy. Our model allows us to decouple the inherent job size (X) from the server-side slowdown (S), where we track both S and X for each job. Analysis within the S&X model is, of course, much more difficult. Nevertheless, we design a policy, Redundant-to- Idle-Queue (RIQ) which is both analytically tractable within the S&X model and has provably excellent performance.

Soundness Proofs for Iterative Deepening

Ben Blum

Carnegie Mellon University Parallel Data Lab Technical Report CMU-PDL-16-103, September 6, 2016.

The Iterative Deepening algorithm allows stateless model checkers to adjust preemption points on-the-fly. It uses dynamic data-race detection to avoid necessarily preempting on every shared memory access, and ignores false-positive data race candidates arising from certain heap allocation patterns. An Iterative Deepening test that reaches completion soundly verifies all possible thread interleavings of that test.

Larger-than-Memory Data Management on Modern Storage Hardware for In-Memory OLTP Database Systems

Lin Ma, Joy Arulraj, Sam Zhao, Andrew Pavlo, Subramanya R. Dulloor, Michael J. Giardino, Jeff Parkhurst, Jason L. Gardner, Kshitij Dosh & Col. Stanley Zdonik

DaMoN'16, June 26 - July 01 2016, San Francisco, CA, USA

In-memory database management systems (DBMSs) outperform disk-oriented systems for on-line transaction processing (OLTP) workloads. But this improved performance is only achievable when the database is smaller than the amount of physical memory available in the system. To overcome this limitation, some in-memory DBMSs can move cold data out of volatile DRAM to secondary storage. Such data appears as if it resides in memory with the rest of the database even though it does not. Although there have been several implementations proposed for this type of cold data storage, there has not been a thorough evaluation of the design decisions in implementing this technique, such as policies for when to evict tuples and how to bring them back when they are needed. These choices are further complicated by the varying performance characteristics of different storage devices, including future non-volatile memory technologies. We explore these issues in this paper and discuss several approaches to solve them. We implemented all of these approaches in an in-memory DBMS and evaluated them using five different storage technologies. Our results show that choosing the best strategy based on the hardware

continued on page 29

continued from page 28

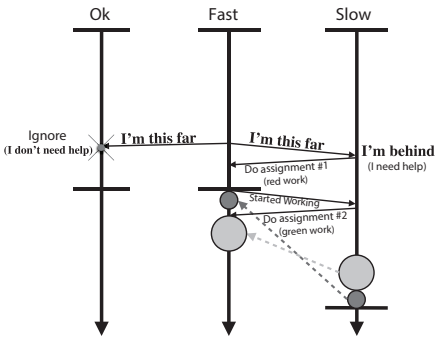
improves throughput by 92–340% over a generic configuration.

Addressing the Straggler Problem for Iterative Convergent Parallel ML

Aaron Harlap, Henggang Cui, Wei Dai, Jinliang Wei Gregory R. Ganger, Phillip B. Gibbons, Garth A. Gibson & Eric P. Xing

ACM Symposium on Cloud Computing 2016. October 5 - 7, Santa Clara. CA.

FlexRR provides a scalable, efficient solution to the straggler problem for iterative machine learning (ML). The frequent (e.g., per iteration) barriers used in traditional BSP-based distributed ML implementations cause every transient slowdown of any worker thread to delay all others. FlexRR combines a more flexible synchronization model with dynamic peer-to-peer reassignment of work among workers to address straggler threads. Experiments with real straggler behavior observed on Amazon EC2 and Microsoft Azure, as well as injected straggler behavior stress tests, confirm the significance of the problem and the effectiveness of FlexRR's solution. Using FlexRR, we consistently observe near-ideal run-times (relative to no performance jitter) across all real and injected straggler behaviors tested.



RapidReassignment example. The middle worker sends progress reports to the other two workers (its helper group). The worker on the left is running at a similar speed, so it ignores the message. The worker on the right is running slower, so it sends a do-this message to re-assign an initial work assignment. Once the faster worker finishes its own work and begins helping, it sends a begun-helping message to the slow worker. Upon receiving this, the slow worker sends a do-this with a follow-up work assignment to the fast worker.

Transparent Offloading and Mapping (TOM): Enabling Programmer-Transparent Near-Data Processing in GPU Systems

Kevin Hsieh, Eiman Ebrahimi, Gwangsun Kim, Niladrish Chatterjee, Mike O'Connor, Nandita Vijaykumar, Onur Mutlu & Stephen W. Keckler

Proceedings of the 43rd International Symposium on Computer Architecture (ISCA), Seoul, South Korea, June 18 - 22, 2016.

Main memory bandwidth is a critical bottleneck for modern GPU systems due to limited off-chip pin bandwidth. 3D-stacked memory architectures provide a promising opportunity to significantly alleviate this bottleneck by directly connecting a logic layer to the DRAM layers with high bandwidth connections. Recent work has shown promising potential performance benefits from an architecture that connects multiple such 3D-stacked memories and offloads bandwidth-intensive computations to a GPU in each of the logic layers. An unsolved key challenge in such a system is how to enable computation offloading and data mapping to multiple 3D-stacked memories without burdening the programmer such that any application can transparently benefit from near-data processing capabilities in the logic layer.

Our paper develops two new mechanisms to address this key challenge. First, a compiler-based technique that automatically identifies code to offload to a logic-layer GPU based on a simple cost-benefit analysis. Second, a software/hardware cooperative mechanism that predicts which memory pages will be accessed by offloaded code, and places those pages in the memory stack closest to the offloaded code, to minimize off-chip bandwidth consumption. We call the combination of these two programmer-transparent mechanisms TOM: Transparent Offloading and Mapping.

Our extensive evaluations across a variety of modern memory-intensive GPU workloads show that, without requiring any program modification, TOM significantly improves performance (by 30% on average, and up to 76%) compared to a baseline GPU system that cannot offload computation to 3D-stacked memories.

Bridging the Archipelago between Row-Stores and Column-Stores for Hybrid Workloads

Joy Arulraj, Andrew Pavlo & Prashanth Menon

SIGMOD'16, June 26 - July 01, 2016, San Francisco, CA, USA

Data-intensive applications seek to obtain trill insights in real-time by analyzing a combination of historical data sets alongside recently collected data. This means that to support such hybrid workloads, database management systems (DBMSs) need to handle both fast ACID transactions and complex analytical queries on the same database. But the current trend is to use specialized systems that are optimized for only one of these workloads, and thus require an organization to maintain separate copies of the database. This adds additional cost to deploying a database application in terms of both storage and administration overhead.

To overcome this barrier, we present a hybrid DBMS architecture that efficiently supports varied workloads on the same database. Our approach differs from previous methods in that we use a single execution engine that is oblivious to the storage layout of data without sacrificing the performance

continued on page 30

continued from page 29

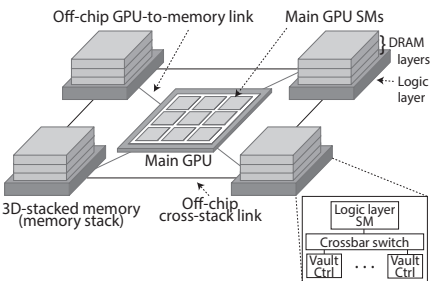
benefits of the specialized systems. This obviates the need to maintain separate copies of the database in multiple independent systems. We also present a technique to continuously evolve the database's physical storage layout by analyzing the queries' access patterns and choosing the optimal layout for different segments of data within the same table. To evaluate this work, we implemented our architecture in an in-memory DBMS. Our results show that our approach delivers up to 3× higher throughput compared to static storage layouts across different workloads. We also demonstrate that our continuous adaptation mechanism allows the DBMS to achieve a near-optimal layout for an arbitrary workload without requiring any manual tuning.

Reducing the Storage Overhead of Main-Memory OLTP Databases with Hybrid Indexes

Huanchen Zhang, Andy Pavlo, David G. Andersen, Michael Kaminsky, Lin Ma & Rui Shen

ACM SIGMOD International Conference on Management of Data 2016 (SIGMOD'16), June 2016.

Using indexes for query execution is crucial for achieving high performance in modern on-line transaction processing databases. For a main-memory database, however, these indexes consume a large fraction of the total memory available and are thus a major source of storage overhead of in-memory databases. To reduce this overhead, we propose using a two-stage index: The first stage ingests all incoming entries and is kept small for fast read and write operations. The index periodically migrates entries from the first stage to the second, which uses a more compact, read-optimized data structure. Our first contribution is hybrid index, a dual-stage index architecture that achieves both space efficiency and high performance. Our second contribution is Dual-Stage Transformation (DST), a set of guidelines for converting any order-preserving index structure into a hybrid index. Our third contribution is applying DST to four popular order-preserving index structures and evaluating them in both standalone microbenchmarks and a full in-memory DBMS using several transaction processing workloads. Our results show that hybrid indexes provide comparable throughput to the original ones while reducing the memory overhead by up to 70%.



Overview of an NDP GPU system.

Achieving One Billion Key-Value Requests Per Second on a Single Server

Sheng Li, Hyeontaek Lim, Victor Lee, Jung Ho Ahn, Anuj Kalia, Michael Kaminsky, David G. Andersen, Seongil O, Sukhan Lee & Pradeep Dubey

IEEE Micro's Top Picks from the Computer Architecture Conferences 2016, May/June 2016. Top Picks 2016!

Distributed in-memory key-value stores (KVSs), such as memcached, have become a critical data serving layer in modern Internet-oriented datacenter infrastructure. Their performance and efficiency directly affect the QoS of web services and the efficiency of datacenters. Traditionally, these systems have had significant overheads from inefficient network processing, OS kernel involvement, and concurrency control. Two recent research thrusts have focused upon improving key-value performance. Hardware-centric research has started to explore specialized platforms including FPGAs for KVSs; results demonstrated an order of magnitude increase in throughput and energy efficiency over stock memcached. Software-centric research revisited the KVS application to address fundamental software bottlenecks and to exploit the full potential of modern commodity hardware; these efforts too showed orders of magnitude improvement over stock memcached.

We aim at architecting high performance and efficient KVS platforms, and start with a rigorous architectural characterization across system stacks over a collection of representative KVS implementations. Our detailed full-system characterization not only identifies the critical hardware/software ingredients for high-performance KVS systems, but also suggests new optimizations to achieve record-setting throughput: 120 million requests per second (MRPS) (167 MRPS when with client-side batching) on a single commodity server. Our system delivers the best performance and energy efficiency (RPS/watt) demonstrated to date with existing KVSs—including the best-published FPGA-based and GPU-based claims. We propose a future manycore platform, and via detailed simulations demonstrate the capability of achieving a billion RPS with a single server constructed following our principles.

Efficient Algorithms with Asymmetric Read and Write Costs

Guy E. Blelloch, Jeremy T. Fineman, Phillip B. Gibbons, Yan Gu & Julian Shun

24th European Symposium on Algorithms (ESA'16). August, 2016.

In several emerging technologies for computer memory (main memory), the cost of reading is significantly cheaper than the cost of writing. Such asymmetry in memory costs poses a fundamentally different model from the RAM for algorithm design. In

continued on page 31

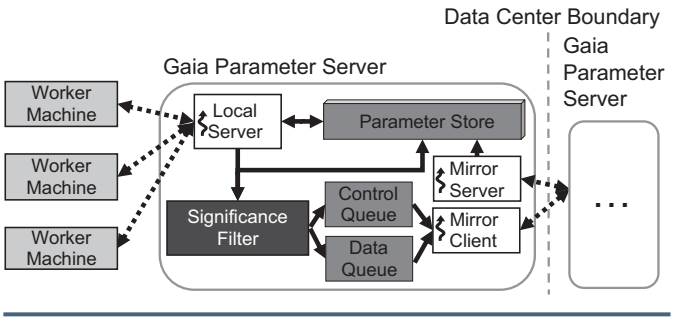
continued from page 30

this paper we study lower and upper bounds for various problems under such asymmetric read and write costs. We consider both the case in which all but $O(1)$ memory has asymmetric cost, and the case of a small cache of symmetric memory. We model both cases using the (M,ω)-ARAM, in which there is a small (symmetric) memory of size M and a large unbounded (asymmetric) memory, both random access, and where reading from the large memory has unit cost, but writing has cost $\omega \gg 1$.

For FFT and sorting networks we show a lower bound cost of $\Omega(\omega n \log_{\omega M} n)$, which indicates that it is not possible to achieve asymptotic improvements with cheaper reads when ω is bounded by a polynomial in M. Moreover, there is an asymptotic gap (of $\min(\omega, \log n)/\log(\omega M)$) between the cost of sorting networks and comparison sorting in the model. This contrasts with the RAM, and most other models, in which the asymptotic costs are the same. We also show a lower bound for computations on an $n \times n$ diamond DAG of $\Omega(\omega n^2/M)$ cost, which indicates no asymptotic improvement is achievable with fast reads. However, we show that for the minimum edit distance problem (and related problems), which would seem to be a diamond DAG, we can beat this lower bound with an algorithm with only $O(\omega n^2/(M \min(\omega^{1/3}, M^{1/2})))$ cost. To achieve this we make use of a "path sketch" technique that is forbidden in a strict DAG computation. Finally, we show several interesting upper bounds for shortest path problems, minimum spanning trees, and other problems. A common theme in many of the upper bounds is that they require redundant computation and a tradeoff between reads and writes.

BIG-LEARNING SYSTEMS MEET CLOUD COMPUTING

continued from page 13



Key components of Gaia.

1% change to the parameter value. With ASP, these insignificant updates to the same parameter within a data center are aggregated (and thus not communicated to other data centers) until the aggregated updates are significant. ASP allows the ML programmer to specify the function and the threshold to determine the significance of updates for each ML algorithm, while providing default configurations for unmodified ML programs. For example, the programmer can specify that all updates that produce more than a 1% change are significant. ASP ensures all significant updates are synchronized across all model copies in a timely manner. It dynamically adapts communication to the available WAN bandwidth between pairs of data centers and uses special selective barrier and mirror clock control messages to ensure algorithm convergence even during a period of sudden fall (negative spike) in available WAN bandwidth.

In contrast to a state-of-the-art communication-efficient synchronization model, Stale Synchronous Parallel (SSP), which bounds how stale (i.e., old) a parameter can be, ASP bounds how inaccurate a parameter can be, compared to the most up-to-date value. Hence, it provides high flexibility in performing (or not performing) updates, as the server can delay synchronization indefinitely as long as the aggregated update remains insignificant.

Experiments with three popular classes of ML algorithms on Gaia prototypes across 11 Amazon EC2 regions show that, compared to two state-of-the-art parameter server systems, Gaia: (1) significantly improves performance, by 1.8–53.5×, (2) has performance within 0.94–1.40× that of running the same ML algorithm on a LAN in a single data center, and (3) significantly reduces the monetary cost of running the same ML algorithm on WANs by 2.6–59.0×. For more information, see [2].

References

[1] Proteus: Agile ML Elasticity through Tiered Reliability in Dynamic Resource Markets. Aaron Harlap, Alexey Tumanov, Andrew Chung, Greg Ganger, Phil Gibbons. ACM European Conference on Computer Systems, 2017 (EuroSys'17), 23rd-26th April, 2017, Belgrade, Serbia.

[2] Gaia: Geo-Distributed Machine Learning Approaching LAN Speeds. Kevin Hsieh, Aaron Harlap, Nandita Vijaykumar, Dimitris Konomis, Gregory R. Ganger, Phillip B. Gibbons, Onur Mutlu. 14th USENIX Symposium on Networked Systems Design and Implementation (NSDI), March 27–29, 2017, Boston, MA.

YEAR IN REVIEW

continued from page 4

- Nandita Vijaykumar presented "Zorua: A Holistic Approach to Resource Virtualization in GPUs" at MICRO '16 in Taipei, Taiwan.
- Papers presented at SoCC '16 in Santa Clara, CA included "Principled Workflow-centric Tracing of Distributed Systems" (Raja R. Sambasivan) and "Addressing the Straggler Problem for Iterative Convergent Parallel ML" (Aaron Harlap).

September 2016

- Kristen Gardner presented "A Better Model for Job Redundancy: Decoupling Server Slowdown and Job Size" at MASCOTS '16 in London, UK.

August 2016

- Yixin Luo proposed his thesis research "Architectural Techniques for Improving NAND Flash Memory Reliability."
- Mor Harchol-Balter presented the keynote presentation on "Queueing with Redundant Requests: A More Realistic Model" at CanQueue 2016 in London, ON.

July 2016

- Alexey Tumanov successfully defended his PhD thesis "Scheduling with Space-Time Soft Constraints in Heterogeneous Cloud Datacenters."
- Gennady Pekhimenko successfully defended his PhD thesis "Practical Data Compression for Modern Memory Hierarchies."
- Phil Gibbons gave the keynote talk on "How Emerging Memory Technologies will Have You Rethinking Algorithm Design" at PODC'16 in Chicago, IL.
- Naama Ben-David presented "Parallel Algorithms for Asymmetric Read-Write Costs" at SPAA '16 in Asilomar, CA.

June 2016

- Anuj Kalia and co-authors won Best Student Paper award at USENIX ATC'16 in Denver, CO for their work on "Design Guidelines for High Performance RDMA Systems."
- Mor Harchol-Balter gave the keynote presentation on "A Better Model for Task Assignment in Server Farms: How Replication can Help" at Sigmetrics '16 in Antibes Juan-les-Pins.
- Phil Gibbons gave the keynote talk on "What's So Special About Big Learning?...A Distributed Systems Perspective" at ICDCS'16 in Nara, Japan.
- Lin Ma presented "Larger-than-Memory Data Management on Modern Storage Hardware for In-Memory OLTP Database Systems" at DaMoN '16 in San Francisco, CA.
- Papers presented at SIGMOD '16 in San Francisco were"Bridging the Archipelago between Row-Stores and Column-Stores for Hybrid Workloads" (Joy Arulraj) and "Reducing the Storage Overhead of Main-Memory OLTP Databases with Hybrid Indexes" (Huanchen Zhang).
- Kevin Hsieh presented "Transparent Offloading and Mapping (TOM): Enabling Programmer-Transparent Near-Data Processing in GPU Systems" at ISCA '16 on Seoul, South Korea.

May 2016

- 18th annual PDL Spring Visit Day.
- "Achieving One Billion Key-Value Requests Per Second on a Single Server" by Sheng Li, Hyeontaek Lim, Victor Lee, Jung Ho Ahn, Anuj Kalia, Michael Kaminsky, David G. Andersen, Seongil O, Sukhan Lee and Pradeep Dubey was chosen as one of IEEE Micro's Top Picks from the Computer Architecture Conferences of 2016.

2016 PDL Workshop and Retreat.