Parallel Data Laboratory
Carnegie Mellon University

FALL UPDATE

PDL PACKET

NEWSLETTER ON PDL ACTIVITIES AND EVENTS • FALL 2016
http://www.pdl.cmu.edu/

PDL CONSORTIUM MEMBERS

Broadcom
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
Uber
Veritas
Western Digital

CONTENTS

THE PDL PACKET

EDITOR
Joan Digney
CONTACTS
Greg Ganger
PDL Director
Bill Courtright
PDL Executive Director
Karen Lindenfelser
PDL Administrative Manager
The Parallel Data Laboratory
Carnegie Mellon University
5000 Forbes Avenue
Pittsburgh, PA 15213-3891
TEL 412-268-6716
FAX 412-268-3010

http://www.pdl.cmu.edu/Publications/

SELECTED RECENT PUBLICATIONS

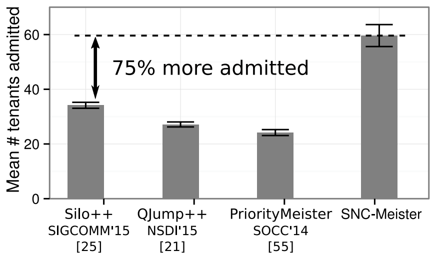
SNC-Meister: Admitting More Tenants with Tail Latency SLOs

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 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.



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.

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 verifica-

continued on page 4

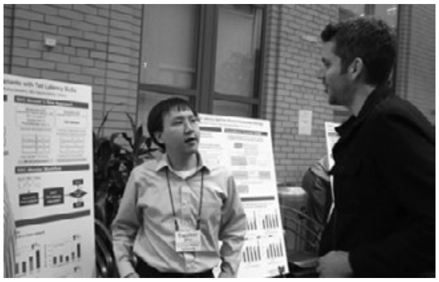
DISSERTATION ABSTRACT: Scheduling with Space-Time Soft Constraints in Heterogeneous Cloud Datacenters

Alexey Tumanov
Carnegie Mellon University, ECE

Ph.D. 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 memoryresident 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.



Timothy Zhu discusses his work on "SNC-Meister: Admitting More Tenants with Tail Latency SLOs" with PDL Alum Elie Krevat (Uber) at the 2016 PDL Spring Visit Day.

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 mis-estimation 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

Ph.D. Defense — July 1, 2016

Although compression has been widely used for decades to reduce file sizes (thereby conserving storage capacity and network bandwidth when transferring files), there has been little to no use of compression within modern memory hierarchies. Why not? Especially 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 could be applied successfully to the memory hierarchy, it could potentially relieve pressure on these bottlenecks by increasing effective capacity, increasing effective bandwidth, and 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

continued on page 3

DEFENSES & PROPOSALS

continued from page 2

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.

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.

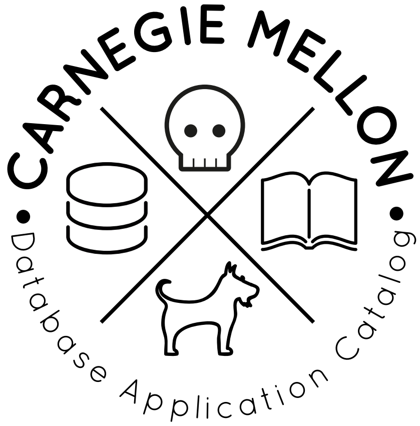
Shasank Chavan (Oracle) explains "Oracle Database In-Memory: The Next Generation" at the 2016 PDL Spring Consortium Speakers Series.

PDL NEWS & AWARDS

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

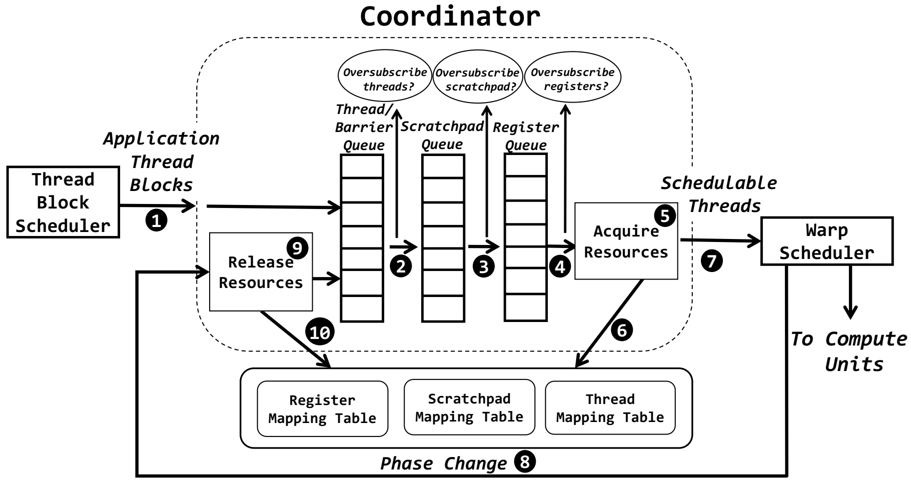
tion of shallow safety properties such as memory and control-flow safety can be easier to automate, but past techniques for automatically verifying machine-code safety have required post-compilation transformations, which can change program behavior. In this work, we automatically verify safety properties 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.

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.



Overview of Zorua in hardware.

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 virtualization. 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.

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

continued on page 5

continued from page 4

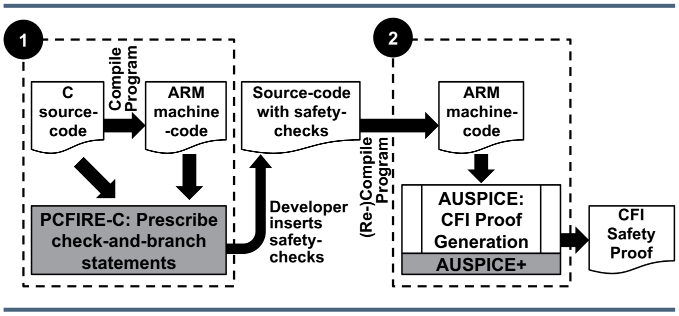
detection, another concurrency testing approach, focuses on suspicious memory access pairs during a single test execution. It avoids concerns of state space size, but may report races that do not lead to observable failures, which jeopardizes a user's willingness to use the analysis.

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.

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.


1
C source-code
Compile Program
ARM machine -code
PCFIRE-C: Prescribe check-and-branch statements
Source-code with safety-checks
Developer inserts safety-checks
2
(Re-)Compile Program
ARM machine-code
AUSPICE: CFI Proof Generation
AUSPICE+
CFI Safety Proof

Overview of PCFIRE's approach. Gray boxes indicate our contributions.

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.

Poster Abstract: BUFS: Towards Bottom-Up Foundational Security for Software in the Internet-of-Things

Jiaqi Tan, Rajeev Gandhi & Priya Narasimhan

1st IEEE/ACM Symposium on Edge Computing (SEC 2016), October 2016.

The Internet-of-Things (IoT) is a rapidly growing phenomenon. While IoT-enabled objects can provide rich features that can improve users' lives, security failures can lead to severe consequences, particularly in safety-critical domains such as medical devices and automobiles. In addition, IoT-enabled objects are often connected to the Internet, increasing their risk for external attacks. Thus, it is important for IoT systems to have strong security guarantees. Some of the security challenges IoT systems face include the need for lightweight cryptographic algorithms and secure communications protocols. In practice, security mechanisms are implemented in a software stack on IoT devices. This software stack needs to (i) provide security mechanisms correctly, and (ii) faithfully execute application logic, without being circumvented by attackers. Software vulnerabilities may allow external attackers to circumvent these security measures: over 250 vulnerabilities were discovered in the top 10 IoT devices in use today in a recent study [1]. We propose BUFS, a bottom-up and foundational approach for verifying the security of the software stack in an IoT system, to provide guarantees for how the software is secure.

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 Parkhusrs, 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

continued on page 6

continued from page 5

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

JamaisVu: Robust Scheduling with Auto-Estimated Job Runtimes

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

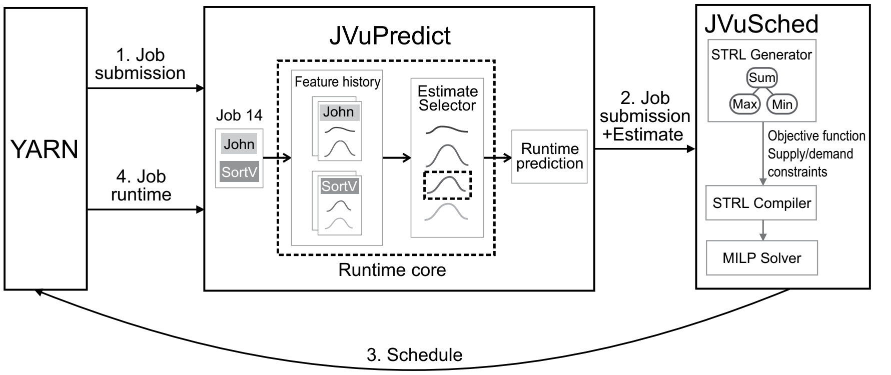
Carnegie Mellon University Parallel Data Laboratory Technical Report CMU-PDL-16-104, September 2016.

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 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.

Soundness Proofs for Iterative Deepening

Ben Blum

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



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

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.

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

continued on page 7

continued from page 6

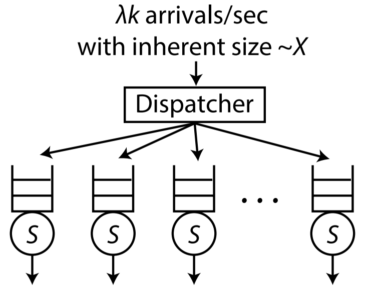
work-efficient (off by the inverse Ackermann function). Our algorithms reveal several interesting techniques for significantly reducing shared memory writes in parallel algorithms without asymptotically increasing the number of shared memory reads.

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.

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.



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 $R(1) = X \cdot S$. 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.

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.

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 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.

Addressing the Straggler Problem for Iterative Convergent Parallel ML

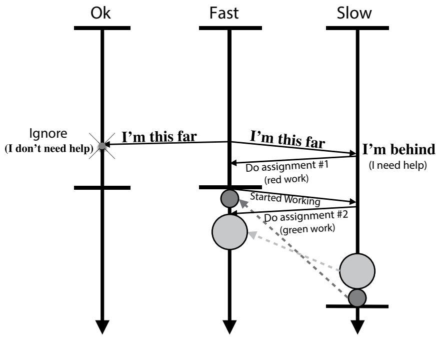
Aaron Harlap, Henggang Cui, Wei Dai, Jinliang Wei Gregory R. Ganger, Phillip B. Gibbons, Garth A. Gibson & Eric P. Xing

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

continued on page 8

continued from page 7



RapidReassignment example. The middle worker sends progress reports to the other two workers (its helpee 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.

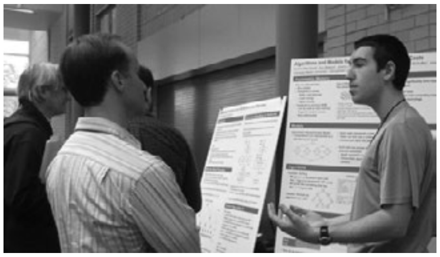
transient slowdown of any worker thread to delay all others. FlexRR combines a more flexible synchronization model with dynamic peer-to-peer reassignment of work among workers to address straggler threads. Experiments with real straggler behavior observed on Amazon EC2 and Microsoft Azure, as well as 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.

PARBOR: An Efficient System-Level Technique to Detect Data-Dependent Failures in DRAM

Samira Khan, Donghyuk Lee & Onur Mutlu

Proceedings of the 45th Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN), Toulouse, France, June 28 - July 1, 2016.

System-level detection and mitigation of DRAM failures offer a variety of system enhancements, such as better reliability, scalability, energy, and performance. Unfortunately, system-level detection is challenging for DRAM failures that depend on the data content of neighboring cells (data-dependent failures). DRAM vendors internally scramble/remap the system-level address space. Therefore, testing data-dependent failures using neighboring system-level addresses does not actually test the cells that are physically adjacent. In this work, we argue that one promising way to uncover data-dependent failures in the system is to determine the location of physically neighboring cells in the system address space. Unfortunately, if done naively, such a test takes 49 days to detect neighboring addresses even in a single memory row, making it infeasible in real systems. We develop PARBOR, an efficient system-level technique that determines the locations of the physically neighboring DRAM cells in the system address space and uses this information to detect data-dependent failures. To our knowledge, this is the first work that solves the challenge of detecting data-dependent failures in DRAM in the presence of DRAM-internal scrambling of system-level addresses. We experimentally demonstrate the effectiveness of PARBOR using 144 real DRAM chips from three major vendors. Our experimental evaluation shows that PARBOR 1) detects neighboring cell locations with only 66-90 tests, a 745-654X reduction compared to the naive test, and 2) uncovers 21.9% more failures compared to a random-pattern test that is unaware of the neighbor cell locations. We introduce a new mechanism that utilizes PARBOR to reduce refresh rate based on the data content of memory locations, thereby improving system performance and efficiency. We hope that our fast and efficient system-level detection technique enables other new ideas and mechanisms that improve the reliability, performance, and energy efficiency of DRAM-based memory systems.



Charles McGuffey discusses his work on "Algorithms and Models for Asymmetric Read-Write Costs" with PDL Alum John Strunk (NetApp).

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 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

continued on page 9

continued from page 8

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.

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

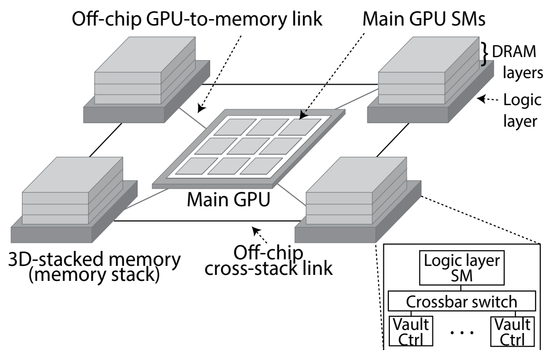
Kevin Hsieh, Eiman Ebrahimi, Gwangsun Kim, Niladrish Chatterjee, Mike O'Connor, Nandita Vijaykumar, Onur Mutlu & Stephen W. Keckler

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.



Overview of an NDP GPU system.

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.

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

continued on page 10

continued from page 9

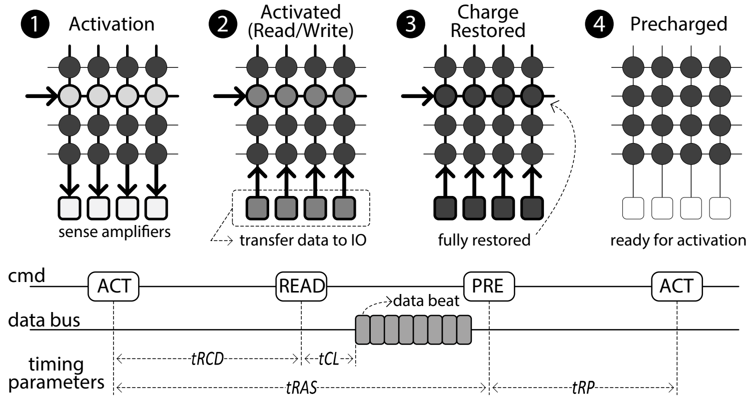
that achieves both space efficiency and high performance. Our second contribution is Dual-Stage Transformation (DST), a set of guidelines for converting any order-preserving index structure into a hybrid index. Our third contribution is applying DST to four popular order-preserving index structures and 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%.

Understanding Latency Variation in Modern DRAM Chips: Experimental Characterization, Analysis, and Optimization

Kevin Chang, Abhijith Kashyap, Hasan Hassan, Samira Khan, Kevin Hsieh, Donghyuk Lee, Saugata Ghose, Gennady Pekhimenko, Tianshi Li & Onur Mutlu

Proceedings of the ACM International Conference on Measurement and Modeling of Computer Systems (SIGMETRICS), Antibes Juan-Les-Pins, France, June 14 - 18, 2016.

Long DRAM latency is a critical performance bottleneck in current systems. DRAM access latency is defined by three fundamental operations that take place within the DRAM cell array: (i) activation of a memory row, which opens the row to perform accesses; (ii) precharge, which prepares the cell array for the next memory access; and (iii) restoration of the row, which restores the values of cells in the row that were destroyed due to activation. There is significant latency variation for each of these operations across the cells of a single DRAM chip due to irregularity in the manufacturing process. As a result, some cells are inherently faster to access, while others are inherently slower. Unfortunately, existing systems do not exploit this variation. The goal of this work is to (i) experimentally characterize and understand the latency variation across cells within a DRAM chip for these three fundamental DRAM operations, and (ii) develop new mechanisms that exploit our understanding of the latency variation to reliably improve performance. To this end, we comprehensively characterize 240 DRAM chips from three major vendors, and make several new observations about latency variation within DRAM. We find that (i) there is large latency variation across the cells for each of the three operations; (ii) variation characteristics exhibit significant spatial locality: slower cells are clustered in certain regions of a DRAM chip; and (iii) the three fundamental operations exhibit different reliability characteristics when the latency of each operation is reduced. Based on our observations, we propose Flexible-LatencY DRAM (FLY-DRAM), a mechanism that exploits latency variation across DRAM cells within a DRAM chip to improve system performance. The key idea of FLY-DRAM is to exploit the spatial locality of slower cells within DRAM, and access the faster DRAM regions with reduced latencies for the fundamental operations. Our evaluations show that FLY-DRAM improves the performance of a wide range of applications by 13.3%, 17.6%, and 19.5%, on average, for each of the three different vendors' real DRAM chips, in a simulated 8-core system. We conclude that the experimental characterization and analysis of latency variation within modern DRAM, provided by this work, can lead to new techniques that improve DRAM and system performance.



Internal DRAM phases, DRAM command/data timelines, and timing parameters to read a cache line.

TierML: Using Tiers of Reliability for Agile Elasticity in Machine Learning

Aaron Harlap, Gregory R. Ganger & Phillip B. Gibbons

Carnegie Mellon University Parallel Data Lab Technical Report CMU-PDL-16-102. May 2016.

The TierML parameter server system for machine learning (ML) enables aggressive exploitation of transient revocable resources to complete model training cheaper and/or faster. 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 (e.g., the Amazon EC2 spot market often provides such resources at a 90% discount). Unlike other parameter server systems, TierML exploits such transient resources, using minimal non-transient resources

continued on page 11

continued from page 10

to efficiently adapt to bulk additions and revocations of transient machines. Our evaluations show that TierML reduces cost by ≈75% relative to non-transient pricing and by 46%-50% relative to using transient resources with checkpointing to address bulk changes, while nearly matching or decreasing running times.

A Case for Hierarchical Rings with Deflection Routing: An Energy-efficient On-chip Communication Substrate

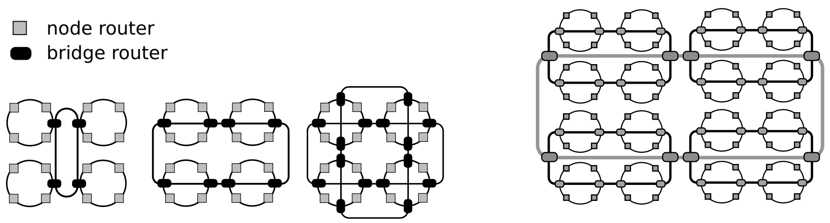
Rachata Ausavarungnirun, Chris Fallin, Xiangyao Yu, Kevin Kai-Wei Chang, Greg Nazario, Reetuparna Das, Gabriel H. Loh & Onur Mutlu

Parallel Computing, Volume 54, May 2016, Pages 29-45, ISSN 0167-8191.

Hierarchical ring networks, which hierarchically connect multiple levels of rings, have been proposed in the past to improve the scalability of ring interconnects, but past hierarchical ring designs sacrifice some of the key benefits of rings by reintroducing more complex in-ring buffering and buffered flow control. Our goal in this paper is to design a new hierarchical ring interconnect that can maintain most of the simplicity of traditional ring designs (i.e., no in-ring buffering or buffered flow control) while achieving high scalability as more complex buffered hierarchical ring designs.

To this end, we revisit the concept of a hierarchical-ring network-on-chip. Our design, called HiRD (Hierarchical Rings with Deflection), includes critical features that enable us to mostly maintain the simplicity of traditional simple ring topologies while providing higher energy efficiency and scalability. First, HiRD does not have any buffering or buffered flow control within individual rings, and requires only a small amount of buffering between the ring hierarchy levels. When inter-ring buffers are full, our design simply deflects flits so that they circle the ring and try again, which eliminates the need for in-ring buffering. Second, we introduce two simple mechanisms that together provide an end-to-end delivery guarantee within the entire network (despite any deflections that occur) without impacting the critical path or latency of the vast majority of network traffic.

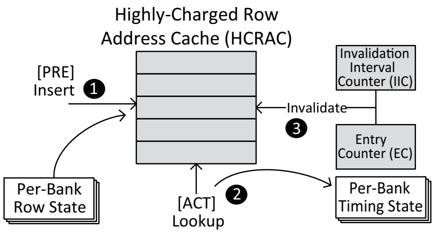
Our experimental evaluations on a wide variety of multiprogrammed and multithreaded workloads and synthetic traffic patterns show that HiRD attains equal or better performance at better energy efficiency than multiple versions of both a previous hierarchical ring design and a traditional single ring design. We also extensively analyze our design's characteristics and injection and delivery guarantees. We conclude that HiRD can be a compelling design point that allows higher energy efficiency and scalability while retaining the simplicity and appeal of conventional ring-based designs.



(a) 4-, 8-, and 16-bridge hierarchical ring designs.

(b) Three-level hierarchy (8x8).

Hierarchical ring design of HiRD.



Components of the ChargeCache Mechanism.

ChargeCache: Reducing DRAM Latency by Exploiting Row Access Locality

Hasan Hassan, Gennady Pekhimenko, Nandita Vijaykumar, Vivek Seshadri, Donghyuk Lee, Oguz Ergin & Onur Mutlu

Proceedings of the 22nd International Symposium on High-Performance Computer Architecture (HPCA), Barcelona, Spain, March 2016.

DRAM latency continues to be a critical bottleneck for system performance. In this work, we develop a low-cost mechanism, called ChargeCache, that enables faster access to recently-accessed rows in DRAM, with no modifications to DRAM chips. Our mechanism is based on the key observation that a recently-accessed row has more charge and thus the following access to the same row can be performed faster. To exploit this observation, we propose to track the addresses of recently-accessed rows in a table in the memory controller. If a later DRAM request hits in that table, the memory controller uses lower timing parameters, leading to reduced DRAM latency. Row addresses are removed from the table after a specified duration to ensure rows that have leaked too much charge are not accessed with lower latency. We evaluate ChargeCache on a wide variety of workloads and show that it provides significant performance and energy benefits for both single-core and multi-core systems.

continued on page 12

continued from page 11

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 bestpublished 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.

Enabling Accurate and Practical Online Flash Channel Modeling for Modern MLC NAND Flash Memory

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

To appear in JSAC Special Issue, 2016.

NAND flash memory is a widely-used storage medium where each of its cells stores data as the threshold voltage of a floating gate transistor. The threshold voltage can shift as the cell wears out, introducing errors and reducing flash lifetime. An accurate model of the threshold voltage distribution can enable mechanisms that improve flash memory reliability and/or performance. Unfortunately, existing models are either not accurate enough or have high computational complexity.

We propose a new, low-complexity flash memory model, built upon a modified version of the Student's t-distribution and the power law, that captures the threshold voltage distribution and predicts future distribution shifts as wear increases. Our model, based upon our experimental characterization of 1X-nm MLC NAND flash chips, achieves 0.68% average modeling error while requiring 4.41x less computation time than the most accurate prior model (with negligible decrease in accuracy). Our model also predicts future threshold voltage distribution shifts with a 2.72% modeling error.

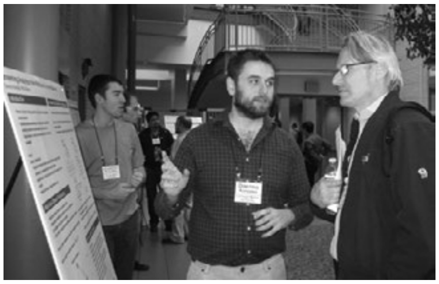
Our model can be used online to enable several applications in the flash controller. We demonstrate two example applications, which improve flash memory lifetime by 48.9%, and/or enable the flash device to safely sustain 69.9% more write operations than manufacturer specifications.

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.



Dimitris Konomis talks about his research on "Streaming Analytics from Multicore to the Wide-Area" with Michael Gleeson (Oracle) at the 2016 PDL Spring Visit Day.