FALL UPDATE

PDL PACKET

NEWSLETTER ON PDL ACTIVITIES AND EVENTS • FALL 2019
https://www.pdl.cmu.edu/

Parallel Data Lab Receives Computing Cluster from Los Alamos National Lab

Marika Yang

Carnegie Mellon University has received a supercomputer from Los Alamos National Lab (LANL) that will be reconstructed into a computing cluster operated by the Parallel Data Lab (PDL) and housed in Carnegie Mellon's Data Center Observatory. This new computer cluster will augment the existing Narwhal, also from LANL and made up of parts of the decommissioned Roadrunner supercomputer technology, the fastest supercomputer in the world from June 2008 to June 2009.

This new supercomputer, tentatively named Wolf, will be an important part of educating the next generation of computer science professionals, researchers, and educators at Carnegie Mellon. The system was recently retired from LANL's open institutional computing environment and while no longer efficient for simulation science, it still has high value as a training tool and for computer science research. Wolf is made up of 616 computing nodes, each containing two eight-core Intel Xeon Sandy Bridge processors, totaling 9,856 processing cores across the entire cluster. The cluster interconnect is QDR InfiniBand, providing a network that is 30 times faster than Narwhal. Altogether, it will have the capability of about 200 Teraflops, where a Teraflop represents one trillion computations per second.

"Wolf's processing cores are each significantly faster than the previous system, and it consists of about 50 percent more computing nodes," said George Amvrosiadis, assistant research professor of Electrical and Computer Engineering and the Parallel Data Lab (PDL). "We will be retiring the Narwhal nodes. Our experienced PDL team, with Jason Boles leading the installation effort, is doing this gradually to make sure everything works as expected."

continued on page 12

Wolf Cluster: QLogic 640-port InfiniBand Switch

PDL CONSORTIUM MEMBERS

Alibaba Group
Amazon
Datrium
Facebook
Google
Hewlett Packard Enterprise
Hitachi, Ltd.
IBM Research
Intel Corporation
Micron
Microsoft Research
NetApp, Inc.
Oracle Corporation
Salesforce
Samsung Semiconductor
Seagate Technology
Two Sigma

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

September 2019
Abutalib Aghayev Awarded Hima and Jive Graduate Fellowship

Congratulations to Talib on receiving the Hima and Jive Fellowship this year! An anonymous donor established the Hima and Jive Fellowship in Computer Science for International Students in 2012 to support one third-year graduate student annually in the Computer Science Department who has a permanent residence outside the United States, regardless of their national origin. This fellowship is to encourage students to overcome challenges and to have fun doing it. The fellowship is given to one international student in the School of Computer Science annually.

September 2019
Beckmann Earns NSF Early CAREER Award

Nathan Beckmann, an assistant professor in the Computer Science Department, has received a Faculty Early Career Development Award, the NSF's most prestigious award for young faculty members.

Nathan Beckmann, an assistant professor in the Computer Science Department, has received a five-year, roughly $500,000 Faculty Early Career Development (CAREER) Award, the National Science Foundation's most prestigious award for young faculty members.

Beckmann's research interests include computer architecture and performance modeling. The NSF grant will support his work crafting and evaluating a new computer system design that makes accessing data faster and cheaper. Beckmann said more energy efficiency is needed to sustain growth in computing power for machine learning, social networking and robotics.

Applications currently have no control over how data is managed because memory hierarchy is fixed in hardware and hidden from software, resulting in unnecessary data movement. Beckmann's project will develop a new hardware-software co-design, wherein the operating system and hardware will collaboratively schedule tasks and data to improve efficiency.

Beckmann will involve high school, undergraduate and graduate students in research. He will also organize research workshops for undergraduate women and a summer internship program for underrepresented minorities.

Beckmann earned his master's degree and Ph.D. from the Massachusetts Institute of Technology, where he spent one year post-doc in the Computer Science and Artificial Intelligence Laboratory.

-- SCS News - September 12, 2019

September 2019
Welcome Hazel!

Jun Woo's daughter Hazel Park was born on September 28, 2019 at 8:24 p.m. She was 7 pounds 11 ounces and 20 inches long. All are doing well at home!

July 2019
Summer Internships

Several PDL grad students interned with our sponsor companies this summer. Ankush Jain interned at LANL over the summer, and Daniel Wong and Giulio Zhou worked at Google.

July 2019
Welcome Chester!

Jason Boles, his wife Chien-Chiao, and big brother Jonas welcomed Chester at 1:44 am July 10, 2019. He weighed in at 7lbs 4oz. and was 20 inches long.

June 2019
George Amvrosiadis Reports on the Future of Storage

George Amvrosiadis led 33 scientists from academia, industry, and federal agencies in the compilation of a report on future storage research for the National Science Foundation (NSF). Their Data Storage Research Vision 2025 recommends effort in four key areas for innovative research and education: enhancing cloud and edge computing I/O infrastructures; designing storage for emerging AI applications; rethinking the storage systems abstractions in service of for new and innovative applications; and redesigning storage systems for emerging hardware.

April 2019
Schwedock receives NSF Graduate Research Fellowship

Brian Schwedock, an electrical and

continued on page 3

PDL NEWS & AWARDS

continued from page 2

computer engineering Ph.D. student, has received the prestigious National Science Foundation (NSF) Graduate Research Fellowship for his work in computer architecture and computer systems with a focus on caching.

Schwedock's current project improves the performance and energy efficiency of chip-multiprocessors in data centers. Data centers waste significant amounts of hardware, energy, and capital by isolating applications with different priorities, specifically latency-critical applications and batch applications.

"My project proposes an operating system runtime which reduces this waste by intelligently sharing hardware caches among these different applications," says Schwedock. "Our results show major improvements in performance and energy efficiency for low priority batch applications while still meeting strict deadlines required by high priority latency-critical applications."

The NSF Graduate Research Fellowship Program recognizes and supports outstanding graduate students in NSF-supported science, technology, engineering, and mathematics disciplines who are pursuing research-based Master's and doctoral degrees at accredited United States institutions.

Schwedock is advised by Nathan Beckmann, assistant professor in the Computer Science Department.

Congratulations are also due to Giulio Zhou, who received an honorable mention for the NSF Graduate Research Fellowship Program this year.

-- ECE News and Events - April 18, 2019

RECENT PUBLICATIONS

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

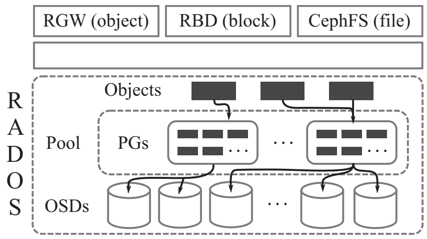
File Systems Unfit as Distributed Storage Backends: Lessons from 10 Years of Ceph Evolution

Abutalib Aghayev, Sage Weil, Michael Kuchnik, Mark Nelson, Gregory R. Ganger & George Amvrosiadis

SOSP '19, October 27–30, 2019, Huntsville, ON, Canada.

For a decade, the Ceph distributed file system followed the conventional wisdom of building its storage backend on top of local file systems. This is a preferred choice for most distributed file systems today because it allows them to benefit from the convenience and maturity of battle-tested code. Ceph's experience, however, shows that this comes at a high price. First, developing a zero-overhead transaction mechanism is challenging. Second, metadata performance at the local level can significantly affect performance at the distributed level. Third, supporting emerging storage hardware is painstakingly slow.

Ceph addressed these issues with BlueStore, a new backend designed to run directly on raw storage devices. In only two years since its inception, BlueStore outperformed previous established backends and is adopted by 70% of users in production. By running in user space and fully controlling the I/O stack, it has enabled space-efficient metadata and data checksums, fast overwrites of erasure-coded data, inline compression, decreased performance variability, and avoided a series of performance pitfalls of local file systems. Finally, it makes the adoption of backwards-incompatible storage hardware possible, an important trait in a changing storage landscape that is learning to embrace hardware diversity.



High-level depiction of Ceph's architecture. A single pool with 3× replication is shown. Therefore, each placement group (PG) is replicated on three OSDs.

Parity Models: Erasure-Coded Resilience for Prediction Serving Systems

Jack Kosaian, K. V. Rashmi & Shivaram Venkataraman

SOSP '19, October 27–30, 2019, Huntsville, ON, Canada.

Machine learning models are becoming the primary workhorses for many applications. Services deploy models through prediction serving systems that take in queries and return predictions by performing inference on models. Prediction serving systems are commonly run on many machines in cluster settings, and thus are prone to slowdowns and failures that inflate tail latency. Erasure coding is a popular technique for achieving resource-efficient resilience to data unavailability in storage and communication systems. However, existing approaches for imparting erasure-coded resilience to distributed computation apply only to a severely limited class of functions, precluding their use for many serving workloads, such as neural network inference. We introduce parity models,

continued on page 8

DISSERTATION ABSTRACT: Enhancing Programmability, Portability, and Performance with Rich Cross-Layer Abstractions

Nandita Vijaykumar
Carnegie Mellon University, ECE

PhD Defense — October 11, 2019

Programmability, performance portability, and resource efficiency have emerged as critical challenges in harnessing complex and diverse architectures today to obtain high performance and energy efficiency. While there is abundant research, and thus significant improvements, at different levels of the stack that address these very challenges, in this thesis, we observe that we are fundamentally limited by the interfaces and abstractions between the application and the underlying system/hardware—specifically, the hardware-software interface. The existing narrow interfaces poses two critical challenges. First, significant effort and expertise are required to write high-performance code to harness the full potential of today's diverse and sophisticated hardware. Second, as a hardware/system designer, architecting faster and more efficient systems is challenging as the vast majority of the program's semantic content gets lost in translation with today's application-system interfaces. Moving towards the future, these challenges in programmability and efficiency will be even more intractable as we architect increasingly heterogeneous and sophisticated systems.

This thesis makes the case for rich low-overhead cross-layer abstractions as a highly effective means to address the above challenges. These abstractions are designed to communicate higher-level program information from the application to the underlying system and hardware in a highly efficient manner, requiring only minor additions to the existing interfaces. In doing so, they enable a rich space of hardware-software cooperative mechanisms to optimize for performance. We propose 4 different approaches to designing richer abstractions between the application, system software, and hardware architecture in different contexts to significantly improve programmability, portability, and performance in CPUs and GPUs: (i) Expressive Memory: A unifying cross-layer abstraction to express and communicate higher-level program semantics from the application to the underlying system/architecture to enhance memory optimization; (ii) The Locality Descriptor: A cross-layer abstraction to express and exploit data locality in GPUs; (iii) Zorua: A framework to decouple the programming model from management of on-chip resources and parallelism in GPUs; (iv) Assist Warps: A helper-thread abstraction to dynamically leverage underutilized compute/memory bandwidth in GPUs to perform useful work. In this thesis, we present each concept and describe how communicating higher-level program information from the application can enable more intelligent resource management by the architecture and system software to significantly improve programmability, portability, and performance in CPUs and GPUs.

Oracle Autonomous Database:
Sit-back, relax and let Oracle do the driving!
ORACLE

Aurosish Mishra, Oracle, talks about Oracle's Autonomous Database at the special SDI/Visit Day Industry Seminar.

DISSERTATION ABSTRACT: Memory-Efficient Search Trees for Database Management Systems

Huanchen Zhang
Carnegie Mellon University, SCS

PhD Defense — October 4, 2019

The growing cost gap between DRAM and storage together with increasing database sizes means that database management systems (DBMSs) now operate with a lower memory to storage size ratio than before. On the other hand, modern DBMSs rely on in-memory search trees (e.g., indexes and filters) to achieve high throughput and low latency. These search trees, however, consume a large portion of the total memory available to the DBMS. This dissertation seeks to address the challenge of building compact yet fast in-memory search trees to allow more efficient use of memory in data processing systems. We first present techniques to obtain maximum compression on fast read-optimized search trees. We identified sources of memory waste in existing trees and designed new succinct data structures to reduce the memory to the theoretical limit. We then introduce ways to amortize the cost of modifying static data structures with bounded and modest cost in performance and space. Finally, we approach the search tree compression problem from an orthogonal direction by building a fast order-preserving key compressor. Together, these three pieces form a practical recipe for achieving memory-efficiency in search trees and in DBMSs.

DISSERTATION ABSTRACT: Machine Learning Systems for Highly-Distributed and Rapidly-Growing Data

Kevin Hsieh
Carnegie Mellon University, ECE

PhD Defense — September 5, 2019

The usability and practicality of any machine learning (ML) applications are largely influenced by two critical but hard-to-attain factors: low latency and low cost. Unfortunately, achieving low latency and low cost is very challenging when ML depends on real-world data that are highly distributed and rapidly growing (e.g., data collected by mobile phones and video cameras all over the

continued on page 5

continued from page 4

world). Such real-world data pose many challenges in communication and computation. For example, when training data are distributed across data centers that span multiple continents, communication among data centers can easily overwhelm the limited wide-area network bandwidth, leading to prohibitively high latency and high cost.

In this dissertation, we demonstrate that the latency and cost of ML on highly-distributed and rapidly-growing data can be improved by one to two orders of magnitude by designing ML systems that exploit the characteristics of ML algorithms, ML model structures, and ML training/serving data. We support this thesis statement with three contributions. First, we design a system that provides both low-latency and low-cost ML serving (inferencing) over large-scale and continuously-growing datasets, such as videos. Second, we build a system that makes ML training over geo-distributed datasets as fast as training within a single data center. Third, we present a first detailed study and a system-level solution on a fundamental and largely overlooked problem: ML training over non-IID (i.e., not independent and identically distributed) data partitions (e.g., facial images collected by cameras will reflect the demographics of each camera's location).

DISSERTATION ABSTRACT: Efficient Remote Procedure Calls for Datacenters

Anuj Kalia
Carnegie Mellon University, SCS

PhD Defense — August 30, 2019

Datacenter network latencies are approaching their microsecond-scale speed-of-light limit, and network bandwidths continue to grow beyond 100 Gbps. These improvements bear rethinking the design of communication-intensive distributed systems for datacenters, whose performance has historically been limited by slow networks. With the slowing down of Moore's law, a popular approach is to redesign distributed systems to use custom network hardware devices and technologies—smart network cards (NICs), lossless networks, programmable NICs, and programmable switches—that offload communication or data access from commodity CPUs.

In this dissertation, we show that we can continue to use end-to-end communication mechanisms to build high-performance distributed systems with commodity hardware in modern datacenters, i.e., we bring the speed of fast networks to distributed systems without requiring an expensive redesign with custom hardware. We show that the ubiquitous Remote Procedure Call (RPC) communication mechanism, when rearchitected specially for the capabilities of modern commodity datacenter hardware, is a fast, scalable, flexible, and simple communication choice for distributed systems. We make three contributions. First, we present a detailed analysis of datacenter communication hardware—ranging from the peripheral bus that connects CPUs to NICs, to the datacenter's switched network—that informs our choice of the communication mechanism. Second, we lay out the advantages of RPCs over in-network offloads through the design and evaluation of two new systems, a key-value store called HERD, and a distributed transaction processing system called FaSST. Third, we combine the lessons learned from the first two steps with new insights about datacenter packet loss and congestion control to create a new RPC library called eRPC, and show how existing distributed system codebases perform well over eRPC. In many cases, these systems substantially outperform offloads because they use less communication, and their end-to-end design provides flexibility and simplicity.

Abutalib Aghayev discusses his work on Reconciling LSM-Trees with Modern Hard Drives using BlueFS with Jacob Strauss of Amazon Web Services.

DISSERTATION ABSTRACT: Data Structure Engineering for High Performance Software Packet Processing

Dong Zhou
Carnegie Mellon University, SCS

PhD Defense — July 31, 2019

Compared with using specialized hardware, software packet processing on general-purpose hardware provides extensibility and programmability. From software routers to virtual switches to Network Function Virtualization, we are seeing increasing applications of software-based packet processing. However, software-based solutions often face performance challenges, primarily because general-purpose CPUs are not optimized for processing network packets.

We observed that for a wide range of packet processing applications, performance is bottlenecked by one or more data structures. Therefore, this thesis tackles the performance of software packet processing by optimizing the main data structures of the application. To demonstrate the effectiveness of our approach, we examined three applications: Ethernet forwarding, LTE-to-Internet gateway and virtual switches. For each application, we propose algorithmic refinements and engineering principles to improve its main data structures, including:

- » A concurrent, read-optimized hash table for x86 platform
- » An extremely compact data structure for set separation

continued on page 6

continued from page 5

- A new cache design that balances between cache hit rate and lookup latency.

In all three applications, we are able to achieve higher performance than existing solutions. For example, our Ethernet switch can saturate the maximum number of packets achievable by the underlying hardware, even with one billion FIB entries in the forwarding table.

THESIS PROPOSAL: Practical Mechanisms for Reducing Processor-Memory Data Movement in Modern Workloads

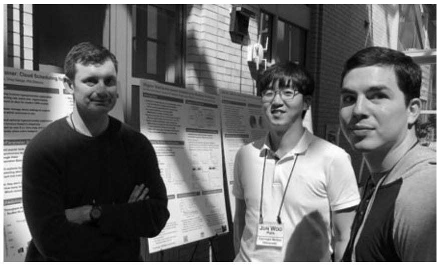
Amirali Boroumand, ECE
September 13, 2019

Data movement between the memory system and computation units is one of the most critical challenges in designing high performance and energy-efficient computing system. The high cost of data movement is forcing architects to rethink the fundamental design of computer systems. Recent advances in memory design enable the opportunity for architects to avoid unnecessary data movement by performing Processing-In-Memory (PIM), also known as Near-Data Processing (NDP). While PIM can allow many data-intensive applications to avoid moving data from memory to the CPU, it introduces new challenges for system architects and programmers. Our goal in this thesis is to make PIM effective and practical in conventional computing systems. Toward this end, this thesis presents three major directions: (1) examining the suitability of PIM across key workloads, (2) addressing major system challenges for adopting PIM in computing systems, and (3) re-designing applications aware of PIM capability. As preliminary steps, we have already developed and evaluated two mechanisms related to the first two major directions of our thesis. Our first preliminary work aimed to identify important primitives for PIM by investigating the suitability of PIM across key Google consumer workloads. Our second preliminary work, called CoNDA, aimed to address the coherence challenge by proposing an efficient cache coherence support for PIM. As for future proposed works, we aim to explore how we can redesign applications aware of PIM capability using software-hardware co-design approach. As our first proposed work, we propose to re-design emerging modern hybrid databases aware of PIM capability to enable real-time analysis. For the second proposed work, we propose a hardware-software co-design approach aware of PIM for mobile machine learning applications to enable energy efficient and high performance inference execution. If successful, we expect the mechanisms proposed by this thesis to make PIM more effective and practical in computing systems.

THESIS PROPOSAL: Accelerating Genome Sequence Analysis via Efficient Hardware-Algorithm Co-Design

Damla Senol, ECE
September 6, 2019

Genome sequence analysis has the potential to enable significant advancements in areas such as personalized medicine, evolution, and forensics. However, effectively leveraging genome sequencing as a tool requires very high computational power. As prior works have shown, many of the core steps in genome sequencing are bottlenecked by the current capabilities of computer systems, as these steps must process a large amount of data. Our goals in this proposal are to (1) analyze the multiple steps and the associated tools in the genome sequence analysis pipeline, (2) expose the tradeoffs between accuracy, performance, memory usage and scalability, and (3) co-design efficient algorithms along with scalable and energy-efficient hardware accelerators for the key bottleneck steps of the pipeline to enable faster genome sequence analysis. To this end, we first describe our first work, which we 1) analyze the multiple steps and the associated tools in the genome sequence analysis pipeline, and 2) expose the tradeoffs between accuracy, performance, memory usage, and scalability. Next, we describe our second work, BitMAC, an in-memory accelerator for generic approximate string matching algorithms that includes specialized support for the read mapping and read-to-read overlap finding steps of the pipeline. For our future work, we propose to explore four new works. In the first work, we propose to replace the PIM core of BitMAC-TB for the traceback step of the read alignment with a new accelerator design, to further increase the efficiency of BitMAC. In the second work, we propose to enhance the algorithmic contributions of BitMAC and provide more functionality. In the third work, we propose to design an accelerator for generic graph processing algorithms that includes specialized support for the assembly step of the genome sequence analysis pipeline. In the fourth work, we propose to design an accelerator for recurrent neural networks that includes specialized support for the basecalling step. We aim to develop and evaluate a variety of acceleration mechanisms, including specialized accelerators, in-memory processing engines, and SIMD architectures. We hope that this research will demonstrate that genome sequence analysis can be accelerated by co-designing scalable and energy-efficient customized accelerators along with efficient algorithms for different steps of the analysis pipeline. We also hope that this research will inspire future work in co-designing software and specialized hardware for emerging application domains.



Jim Cipar, PDL alum, now with Facebook, visits with Jun Woo Park and Aaron Harlap at the 2019 PDL Spring Visit Day.

continued on page 7

continued from page 6

THESIS PROPOSAL: Efficient Direct Access NVM Storage Redundancy

Rajat Kateja, ECE
Date: July 9th 2019

Non-volatile memory (NVM) technologies combine DRAM-like performance with disk-like durability. Direct-access (DAX) to NVM enables load/store access to persistent data and eliminates system software overheads. Production DAX NVM storage deployments will demand conventional system-redundancy mechanisms like per-page checksums and cross-page parity. Maintaining system-redundancy with DAX is challenging because of two reasons. First, system software bypass makes it challenging to identify data accesses that should trigger system-redundancy updates and verification. Second, incongruence in the DAX granularity and typical system-redundancy granularity increases the performance overhead. In this work, we present solutions to provide low-overhead DAX NVM storage redundancy by delaying data coverage or leveraging hardware offload for synchronous redundancy maintenance.

THESIS PROPOSAL: Efficiently Adopting Zone Devices in Distributed Storage

Abutalib Aghayev, SCS
June 24, 2019

Distributed storage systems, such as cluster and parallel file systems and distributed object stores, have conventionally relied on general-purpose local file systems as storage backends. So far, this convention has delivered reasonable performance, precluding questions on the suitability of file systems as distributed storage backends.

Recent developments in the storage hardware targeted at data centers, however, present a challenge for this convention. Solid-state drives (SSDs) are abandoning the flash translation layer to achieve predictable performance and low tail latency. Hard disk drives (HDDs) are adopting shingled magnetic recording for higher capacity at low cost. Most importantly, these data center SSDs and HDDs are evolving to use the same new backward-incompatible zone interface. Adopting these devices is problematic for most file systems because file systems heavily depend on the venerable block interface and carry the legacy of decades-old design from the era of small drives and single-node operating systems.

Our thesis is that to achieve the low cost and predictable performance offered by zone devices, distributed storage systems should abandon file systems as storage backends and implement specialized backends from scratch that allow them to quickly and effectively leverage the benefits of zone devices.

In this proposal, we present the following evidence to support our thesis. We show that using file systems on HDDs with a translation layer has high garbage collection cost: even on a sequential workload, the overhead can be up to 40%. We perform a longitudinal study of storage backends in Ceph—a widely-used distributed storage system—and show that essential services, such as transactions, can be up to 80% faster when implemented directly on a raw device, compared to when implemented on top of file systems. We propose techniques for adapting BlueStore, a Ceph backend implemented on raw devices, to work effectively on top of zone devices.

THESIS PROPOSAL: Distributed Metadata and Streaming Data Indexing as Scalable Filesystem Services

Qing Zheng, SCS
June 14, 2019

As people build larger and more powerful supercomputers, the sheer size of future machines will bring unprecedented levels of concurrency. For applications that write one file per process, increased concurrency will cause more files to be accessed simultaneously and this requires the metadata information of these files to be managed more efficiently. An important factor preventing existing HPC filesystems from being able to more efficiently absorb filesystem metadata mutations is the continued use of a single, globally consistent filesystem namespace to serve all applications running on a single computing environment. Having a shared filesystem namespace accessible from anywhere in a computing environment has many welcome benefits, but it increases each application process' communication with the filesystem's metadata servers for ordering concurrent filesystem metadata changes. This is especially the case when all the metadata synchronization and serialization work is coordinated by a small, fixed set of filesystem metadata servers as we see in many HPC platforms today. Since scientific applications are typically self-coordinated batch programs, the first theme of this thesis is about taking advantage of knowledge about the system and scientific applications to drastically reduce, and in extreme cases, remove unnecessary filesystem metadata synchronization and serialization, enabling HPC applications to better enjoy the increasing level of concurrency in future HPC platforms.

Overcoming filesystem metadata bottlenecks during simulation I/O is important. Achieving efficient analysis of large-scale simulation output is an even more important enabler for fast scientific discovery. With future machines, simulations' output will only become larger and more detailed than it is today. To prevent analysis queries from experiencing excessive I/O delays, the simulation's output must be carefully reorganized for efficient retrieval. Data reorganization is necessary because simulation output is not always written in the optimal order for analysis queries. Data reorganization can be prohibitively time-consuming when its process requires data to be readback from storage in large volumes. The second theme of this thesis is about leveraging idle CPU cycles on the compute nodes of an application to perform data reorganization and indexing, enabling data to be transformed to a read-optimized format without undergoing expensive readbacks.

continued on page 12

continued from page 3

a new approach for enabling erasure-coded resilience in prediction serving systems. A parity model is a neural network trained to transform erasure-coded queries into a form that enables a decoder to reconstruct slow or failed predictions. We implement parity models in ParM, a prediction serving system that makes use of erasure-coded resilience. ParM encodes multiple queries into a "parity query," performs inference over parity queries using parity models, and decodes approximations of unavailable predictions by using the output of a parity model. We showcase the applicability of parity models to image classification, speech recognition, and object localization tasks. Using parity models, ParM reduces the gap between 99.9th percentile and median latency by up to 3.5×, while maintaining the same median. These results display the potential of parity models to unlock a new avenue to imparting resource-efficient resilience to prediction serving systems.

Processing-in-Memory: A Workload-Driven Perspective

S. Ghose, A. Boroumand, J. S. Kim, J. Gómez-Luna & O. Mutlu

To appear in IBM Journal of Research and Development (JRD), November 2019.

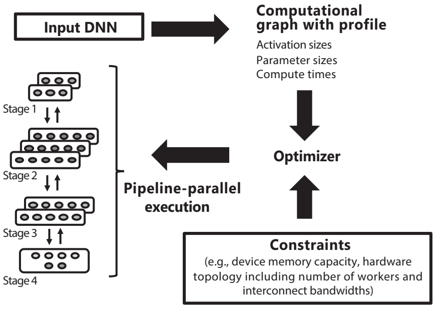
Many modern and emerging applications must process increasingly large volumes of data. Unfortunately, prevalent computing paradigms are not designed to efficiently handle such large-scale data: the energy and performance costs to move this data between the memory subsystem and the CPU now dominate the total costs of computation. This forces system architects and designers to fundamentally rethink how to design computers. Processing-in-memory (PIM) is a computing paradigm that avoids most data movement costs by bringing computation to the data. New opportunities in modern memory systems are enabling architectures that can perform varying degrees of processing inside the memory subsystem. However, there are many practical system-level issues that must be tackled to construct PIM architectures, including enabling workloads and programmers to easily take advantage of PIM. This article examines three key domains of work towards the practical construction and widespread adoption of PIM architectures. First, we describe our work on systematically identifying opportunities for PIM in real applications, and quantify potential gains for popular emerging applications (e.g., machine learning, data analytics, genome analysis). Second, we aim to solve several key issues on programming these applications for PIM architectures. Third, we describe challenges that remain for the widespread adoption of PIM.

PipeDream: Generalized Pipeline Parallelism for DNN Training

Deepak Narayanan, Aaron Harlap, Amar Phanishayee, Vivek Seshadri, Nikhil R. Devanur, Gregory R. Ganger, Phillip B. Gibbons & Matei Zaharia

SOSP '19, October 27–30, 2019, Huntsville, ON, Canada.

DNN training is extremely time-consuming, necessitating efficient multi-accelerator parallelization. Current approaches to parallelizing training primarily use intra-batch parallelization, where a single iteration of training is split over the available workers, but suffer from diminishing returns at higher worker counts. We present PipeDream, a system that adds inter-batch pipelining to intra-batch parallelism to further improve parallel training throughput, helping to better overlap computation with communication and reduce the amount of communication when possible. Unlike traditional pipelining, DNN training is bi-directional, where a forward pass through the computation graph is followed by a backward pass that uses state and intermediate data computed during the forward pass. Naïve pipelining can thus result in mismatches in state versions used in the forward and backward passes, or excessive pipeline flushes and lower hardware efficiency. To address these challenges, PipeDream versions model parameters for numerically correct gradient computations, and schedules forward and backward passes of different minibatches concurrently on different workers with minimal pipeline stalls. PipeDream also automatically partitions DNN layers among workers to balance work and minimize communication. Extensive experimentation with a range of DNN tasks, models, and hardware configurations shows that PipeDream trains models to high accuracy up to 5.3× faster than commonly used intra-batch parallelism techniques.


Input DNN
Computational graph with profile
Activation sizes
Parameter sizes
Compute times
Optimizer
Constraints
(e.g., device memory capacity, hardware topology including number of workers and interconnect bandwidths)
Pipeline-parallel execution
Stage 1
Stage 2
Stage 3
Stage 4

PipeDream's automated mechanism to partition DNN layers into stages. PipeDream first profiles the input DNN, to get estimates for each layer's compute time and output size. Using these estimates, PipeDream's optimizer partitions layers across available machines, which is then executed by PipeDream's runtime.

Multiversioned Page Overlays: Enabling Faster Serializable Hardware Transactional Memory

Ziqi Wang, Michael A. Kozuch, Todd C. Mowry & Vivek Seshadri

28th Parallel Architecture and Compiler Technologies 2019 (PACT'19), Sept 21-25, 2019, Seattle, WA.

Practical and efficient support for

continued on page 9

continued from page 8

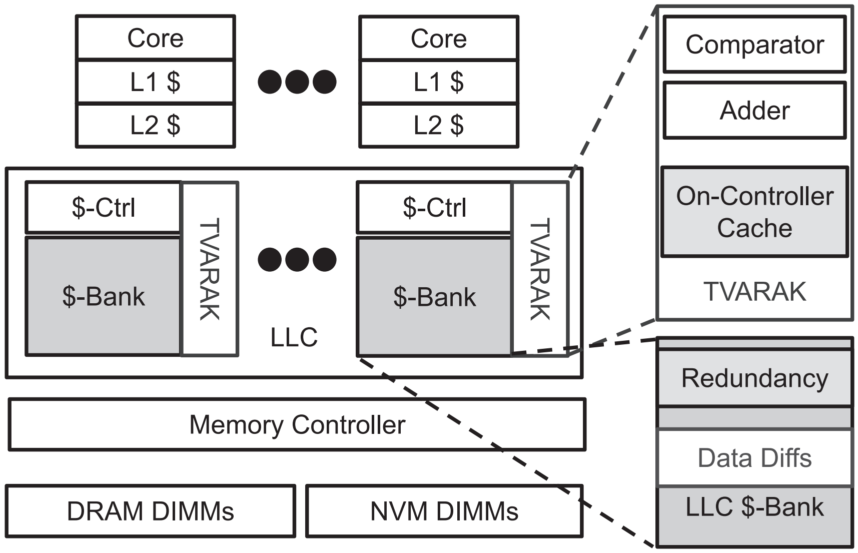
multiversioning memory systems would offer a number of potential advantages, including improving the performance and functionality of hardware transactional memory (HTM). This paper presents a new approach to multiversioning support (Multiversioned Page Overlays) along with a new HTM design that it enables: OverlayTM. Compared with existing HTM designs, OverlayTM takes advantage of multiversioning to reduce unnecessary transaction aborts while providing full serializable semantics (in contrast with multiversioning HTMs that improve performance at the expense of being vulnerable to write skew anomalies). Our performance results demonstrate that OverlayTM is especially advantageous in read-heavy workloads.

TVARAK: Software-Managed Hardware Offload for DAX NVM Storage Redundancy

Rajat Kateja, Nathan Beckmann, & Gregory R. Ganger.

Parallel Data Lab Technical Report CMU-PDL-19-105, Aug 2019.

TVARAK efficiently implements system-level redundancy for direct-access (DAX) NVM storage. Production storage systems complement device-level ECC (which covers media errors) with system-checksums and cross-device parity. This system-level redundancy enables detection of and recovery from data corruption due to device firmware bugs (e.g., reading data from the wrong physical location). Direct access to NVM penalizes software-only implementations of system-level redundancy, forcing a choice between lack of data protection or significant performance penalties. Offloading the update and verification of system-level redundancy to TVARAK, a hardware controller co-located with the last-level cache, enables efficient protection of data from such bugs in memory controller and NVM DIMM firmware. Simulation-based evaluation with seven data-intensive applications shows TVARAK's performance and energy efficiency. For example, TVARAK reduces Redis set-only performance by only 3%, compared to 50% reduction for a state-of-the-art software-only approach.



TVARAK is co-resides with the LLC bank controllers. It includes comparators to identify cache-line that belong to DAX-mapped pages and adders to compute checksums and parity. It includes a small on-controller redundancy cache that is backed by a LLC partition. TVARAK also stores the data diffs to compute checksums and parity.

Compact Filters for Fast Online Data Partitioning

Qing Zheng, Charles D. Cranor, Ankush Jain, Gregory R. Ganger, Garth A. Gibson, George Amvrosiadis, Bradley W. Settlemyer & Gary Grider

IEEE CLUSTER 2019. September 23 - 26, 2019, Albuquerque, New Mexico, USA.

We are approaching a point in time when it will be infeasible to catalog and query data after it has been generated. This trend has fueled research on in-situ data processing (i.e. operating on data as it is streamed to storage). One important example of this approach is in-situ data indexing. Prior work has shown the feasibility of indexing at scale as a two-step process. First, one partitions data by key across the CPU cores of a parallel job. Then each core indexes its subset as data is persisted. Online partitioning requires transferring data over the network so that it can be indexed and stored by the core responsible for the data. This approach is becoming increasingly costly as new computing platforms emphasize parallelism instead of individual core performance that is crucial for communication libraries and systems software in general. In addition to indexing, scalable online data partitioning is also useful in other contexts such as load balancing and efficient comp!

We present FilterKV, an efficient data management scheme for fast online data partitioning of key-value (KV) pairs. FilterKV reduces the total amount of data sent over the network and to storage. We achieve this by: (a) partitioning pointers to KV pairs instead of the KV pairs themselves and (b) using a compact format to represent and store KV pointers. Results from LANL show that FilterKV can reduce total write slowdown (including partitioning overhead) by up to 3x across 4096 CPU cores.

continued on page 10

continued from page 9

STRADS-AP: Simplifying Distributed Machine Learning Programming without Introducing a New Programming Model

Jin Kyu Kim, Abutalib Aghayev, Garth A. Gibson & Eric P. Xing

Proceedings of the 2019 USENIX Annual Technical Conference, July 10–12, 2019 • Renton, WA.

It is a daunting task for a data scientist to convert sequential code for a Machine Learning (ML) model, published by an ML researcher, to a distributed framework that runs on a cluster and operates on massive datasets. The process of fitting the sequential code to an appropriate programming model and data abstractions determined by the framework of choice requires significant engineering and cognitive effort. Furthermore, inherent constraints of frameworks sometimes lead to inefficient implementations, delivering suboptimal performance.

We show that it is possible to achieve automatic and efficient distributed parallelization of familiar sequential ML code by making a few mechanical changes to it while hiding the details of concurrency control, data partitioning, task parallelization, and fault-tolerance. To this end, we design and implement a new distributed ML framework, STRADS-Automatic Parallelization (AP), and demonstrate that it simplifies distributed ML programming significantly, while outperforming a popular data-parallel framework with a non-familiar programming model, and achieving performance comparable to an ML-specialized framework.

Rateless Codes for Distributed Computations with Sparse Compressed Matrices

Ankur Mallick & Gauri Joshi

IEEE International Symposium on Information Theory (ISIT), July 7-12, 2019, Paris, France.

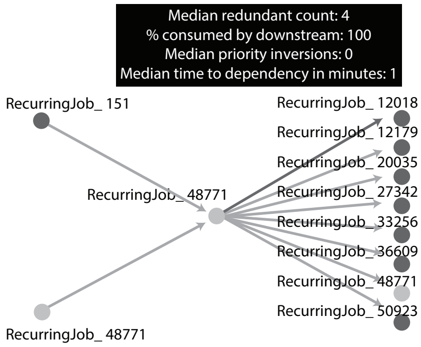
We propose a rateless fountain coding strategy to alleviate the problem of straggling nodes – computing nodes that unpredictably slowdown or fail – in distributed matrix-vector multiplication. Our algorithm generates linear combinations of the m rows of the matrix, and assigns them to different worker nodes, which then perform row-vector products with the encoded rows. The original matrix-vector product can be decoded as soon as slightly more than m row-vector products are collectively completed by the nodes. This strategy enables fast nodes to steal work from slow nodes, without requiring the knowledge of node speeds. Compared to recently proposed fixed-rate erasure coding strategies which ignore partial work done by straggling nodes, rateless codes have a significantly lower overall delay, and a smaller computational overhead.

Peering through the Dark: An Owl's View of Inter-job Dependencies and Jobs' Impact in Shared Clusters

Andrew Chung, Carlo Curino, Subru Krishnan, Konstantinos Karanasos, Panagiotis Garefalakis & Gregory R. Ganger

SIGMOD '19, June 30–July 5, 2019, Amsterdam, Netherlands.

Shared multi-tenant infrastructures have enabled companies to consolidate workloads and data, increasing datasharing and cross-organizational re-use of job outputs. This same resource- and work-sharing has also increased the risk of missed deadlines and diverging priorities as recurring jobs and workflows developed by different teams evolve independently. To prevent incidental business disruptions, identifying and managing job dependencies with clarity becomes increasingly important. Owl is a cluster log analysis and visualization tool that (i) extracts and visualizes job dependencies derived from historical job telemetry and data provenance data sets, and (ii) introduces a novel job valuation algorithm estimating the impact of a job on dependent users and jobs. This demonstration showcases Owl's features that can help users identify critical job dependencies and quantify job importance based on jobs' impact.



Recurring Job dependency graph Displays the target recurring job (center) and its upstream (left) and downstream (right) recurring jobs. Hovering over an upstream/downstream link shows statistics of the dependency

CoNDA: Efficient Cache Coherence Support for Near-Data Accelerators

A. Boroumand, S. Ghose, M. Patel, H. Hassan, B. Lucia, R. Ausavarungnirun, K. Hsieh, N. Hajinazar, K. T. Malladi, H. Zheng & O. Mutlu

Proc. of the International Symposium on Computer Architecture (ISCA), Phoenix, AZ, June 2019.

Specialized on-chip accelerators are widely used to improve the energy efficiency of computing systems. Recent advances in memory technology have enabled near-data accelerators (NDAs), which reside off-chip close to main memory and can yield further benefits than on-chip accelerators. However, enforcing coherence with the rest of the system, which is already a major challenge for accelerators, be-

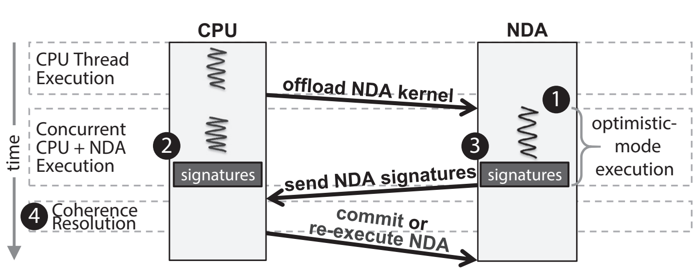
continued on page 11

continued from page 10

comes more difficult for NDAs. This is because (1) the cost of communication between NDAs and CPUs is high, and (2) NDA applications generate a lot of off-chip data movement. As a result, as we show in this work, existing coherence mechanisms eliminate most of the benefits of NDAs. We extensively analyze these mechanisms, and observe that (1) the majority of off-chip coherence traffic is unnecessary, and (2) much of the off-chip traffic can be eliminated if a coherence mechanism has insight into the memory accesses performed by the NDA.

Based on our observations, we propose CoNDA, a coherence mechanism that lets an NDA optimistically execute an NDA kernel, under the assumption that the NDA has all necessary coherence permissions. This optimistic execution allows CoNDA to gather information on the memory accesses performed by the NDA and by the rest of the system. CoNDA exploits this information to avoid performing unnecessary coherence requests, and thus, significantly reduces data movement for coherence.

We evaluate CoNDA using state-of-the-art graph processing and hybrid in-memory database workloads. Averaged across all of our workloads operating on modest data set sizes, CoNDA improves performance by 19.6% over the highest-performance prior coherence mechanism (66.0%/51.7% over a CPU-only/NDA-only system) and reduces memory system energy consumption by 18.0% over the most energy-efficient prior coherence mechanism (43.7% over CPU only). CoNDA comes within 10.4% and 4.4% of the performance and energy of an ideal mechanism with no cost for coherence. The benefits of CoNDA increase with large data sets, as CoNDA improves performance over the highest-performance prior coherence mechanism by 38.3% (8.4x/7.7x over CPU-only/NDA-only), and comes within 10.2% of an ideal no-cost coherence mechanism.



High-level operation of CoNDA. In CoNDA, when an application wants to launch an NDA kernel, the NDA begins executing the kernel in optimistic mode (1). While the NDA kernel executes, all CPU threads continue to execute normally, and never make use of optimistic execution. To gain the insight needed to perform only the necessary coherence requests, CoNDA efficiently tracks the addresses of all NDA reads, NDA writes, and CPU writes during optimistic execution using signatures (2) and (3). Once optimistic execution starts, any NDA data updates are initially flagged as uncommitted. These updates cannot be committed until all necessary coherence requests are performed. When optimistic execution is done, CoNDA attempts to resolve coherence (4).

Understanding Interactions of Workloads and DRAM Types: A Comprehensive Experimental Study

Saugata Ghose, Tianshi Li, Nastaran Hajinazar, Damla Senol Cali & Onur Mutlu

Proc. of the Joint ACM SIGMETRICS/IFIP Performance Conference, Phoenix, AZ, June 2019; To appear in Proceedings of the ACM on Measurement and Analysis of Computing Systems (POMACS), 2019.

It has become increasingly difficult to understand the complex interactions between modern applications and main memory, composed of Dynamic Random Access Memory (DRAM) chips. Manufacturers are now selling and proposing many different types of DRAM, with each DRAM type catering to different needs (e.g., high throughput, low power, high memory density). At the same time, memory access patterns of prevalent and emerging applications are rapidly diverging, as these applications manipulate larger data sets in very different ways. As a result, the combined DRAM–workload behavior is often difficult to intuitively determine today, which can hinder memory optimizations in both hardware and software.

In this work, we identify important families of workloads, as well as prevalent types of DRAM chips, and rigorously analyze the combined DRAM–workload behavior. To this end,we perform a comprehensive experimental study of the interaction between nine different DRAM types and 115 modern applications and multi-programmed workloads. We draw 12 key observations from our characterization, enabled in part by our development of new metrics that take into account contention between memory requests due to hardware design. Notably, we find that (1) newer DRAM technologies such as DDR4 and HMC often do not outperform older technologies such as DDR3, due to higher access latencies and, also in the case of HMC, poor exploitation of locality; (2) there is no single memory type that can effectively cater to all of the components of a heterogeneous system (e.g., GDDR5 significantly outperforms other memories for multimedia acceleration, while HMC significantly outperforms other memories for network acceleration); and (3) there is still a strong need to lower DRAM latency, but unfortunately the current design trend of commodity DRAM is toward higher latencies to obtain other benefits. We hope that the trends we identify can drive optimizations in both hardware and software design. To aid further study,we open-source our extensively-modified simulator, as well as a benchmark suite containing our applications.

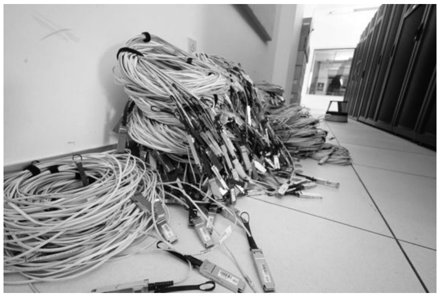
NEW PDL COMPUTING CLUSTER

continued from page 1

In the five years since they received Narwhal from LANL, the researchers of the Parallel Data Lab have developed several projects with the computing cluster in service of educating the world's next thought leaders in several areas of computer science including: scalable storage, cloud computing, machine learning, and operating systems.

"Standing up and operating a reasonably large supercomputer is no small feat," said Brad Settlemyer, senior scientist at LANL. "One of the many reasons that Los Alamos partners with PDL in finding a place for our retired machines is their commitment to providing the staff and resources required to fully utilize this system as an important educational tool."

For example, under the DeltaFS project, a new distributed file system was designed and built enabling scientists to create trillions of files in minutes. With students, faculty, relevant problems, and the right tools PDL has been able to conduct world class research, training, and outcomes such as DeltaFS.

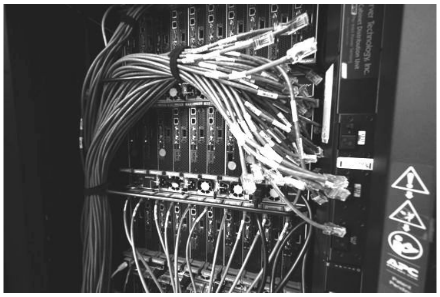


Stacks of InfiniBand QSFP Fiber/Active Optical Cables after being tested.

"The PDL infrastructure enabled us to develop such an ambitious project in-house on Narwhal," said Amvrosiadis. "We were able to use hundreds of nodes to test the scalability of our code, which significantly sped up development and increased our confidence that we could run DeltaFS on Trinity, Los Alamos's fastest supercomputer before we finally did."

Another major benefit of Narwhal, and now Wolf, is having direct access to its hardware on Carnegie Mellon's campus. While there are projects at the Parallel Data Lab that use resources on the cloud to conduct experiments, training future researchers and working towards the future of systems often requires hands-on access to every layer of the machine, from the hardware to all of the software. Having the computing cluster physically on campus allows the researchers to have this control.

The transition from Narwhal to Wolf is currently underway in the Data Center Observatory on the first floor of the Robert Mehrabian Collaborative Innovation Center (CIC). It is a careful and gradual undertaking to ensure all the equipment works as expected, from cables and fans to processors and memory modules, as they can get damaged in the delivery process.



WOLF nodes in the process of being cabled.

The Parallel Data Lab plans to use the new computing cluster for ongoing projects in research areas such as distributed systems, cluster computing, and parallel file systems. Amvrosiadis also anticipates that new projects will become possible with the computing power of Wolf.

"Over the years, we often found ourselves limited by the computational and network capabilities of Narwhal. With Wolf, I expect our experiments will be able to uncover interesting performance trends that are more realistic of contemporary hardware in data centers around the world, making these retired LANL systems a realistic training tool," he said. "Narwhal enabled PDL to conduct training of world-class researchers for many years, and I am looking forward to the research and training that will be made possible by Wolf."

DEFENSES & PROPOSALS

continued from page 7

THESIS PROPOSAL: Efficient ML Training and Inference with Dynamic Hyperparameter Optimization

Angela Jiang, SCS
May 23, 2019

Recent advances in ML have made deep neural networks (DNNs) a fundamental building block of deployed services and applications. But, training DNNs is time-consuming and serving trained DNNs is computationally expensive. Tuning critical hyperparameters improves the efficiency of DNN training and serving, as well as quality of the resulting model. However, almost all of these hyperparameters are generally chosen once at the beginning of training and remain static. We argue that, instead of searching for a single best setting for a hyperparameter, practitioners can achieve superior results by making these hyperparameters adaptive, thus allowing them to fluctuate in response to changing conditions during training and deployment. This has been shown to be true for adaptive learning rates, which are now a standard component of state of the art training regimes. In this thesis we argue that this principle should be extended generally. We provide evidence showing that using runtime information to dynamically adapt hyperparameters that are traditionally static, such as emphasis on individual training examples, augmentation applied to those examples, and the weights updated during transfer learning, can increase the accuracy and efficiency of ML training and inference.