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

CONTENTS

PDL CONSORTIUM MEMBERS

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

PDL PACKET

NEWSLETTER ON PDL ACTIVITIES AND EVENTS • SUMMER 2020
http://www.pdl.cmu.edu/

Parity Models: Erasure-Coded Resilience for Prediction Serving Systems

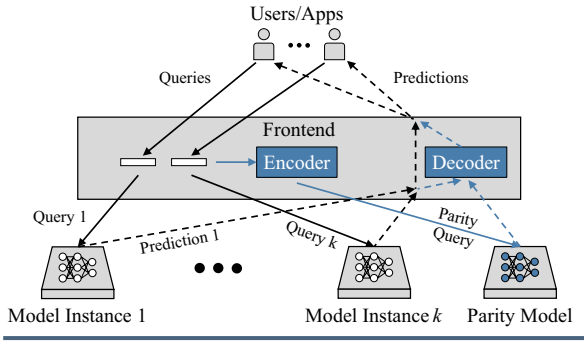
Jack Kosaian, Rashmi Vinayak

Erasure codes have been widely deployed in storage and communication systems to impart resource-efficient resilience against data unavailability. In this work, we describe a new opportunity for using ideas from erasure codes: to reduce tail latency in systems that perform machine learning inference. This raises a number of new challenges compared to traditional uses of erasure codes; most notably, in this new setting, encoded data is computed over. In this work, we show that leveraging machine learning offers the promise of overcoming these challenges to enable the use of erasure codes to reduce tail latency in systems that perform machine learning inference.

Machine learning is widely deployed in production services and user-facing applications. This has increased the importance of inference, the process of returning a prediction from a trained machine learning model. Prediction-serving systems are platforms that host models for inference and deliver predictions for input queries.

To meet the demands of user-facing production services, prediction-serving systems must deliver predictions with low latency (e.g., within tens of milliseconds). Similar to other latency-sensitive services, prediction services must adhere to strict service-level objectives (SLOs). Queries that are not completed by their SLO are often useless to applications. In order to reduce SLO violations, prediction-serving systems must minimize tail latency.

Prediction-serving systems often employ distributed architectures to support high query rates. These systems consist of a frontend, which receives queries and dispatches them to one or more model instances. Model instances perform inference and return predictions. This distributed setup is typically run in large-scale, multi-tenant clusters (e.g., public clouds), where tail latency inflation is a common problem. There are numerous causes of inflated tail latencies in these settings, such as multi-tenancy and resource contention, hardware unreliability and failures, and other complex runtime interactions.



Architecture of a prediction serving system along with components introduced by ParM (shown in blue).

Due to the many causes of tail latency inflation, it is important for mitigations to be agnostic to the cause of slowdown. However, current agnostic approaches for mitigating tail latency inflation either require significant resource overhead by replicating queries, or sacrifice latency by waiting to detect a slowdown or failure before retrying.

continued on page 15

FROM THE DIRECTOR'S CHAIR

GREG GANGER

What a time we're living through. Each year, I use this space to comment on the prior year's research and personal successes of PDL's great students, staff, and faculty. And I will, after this initial paragraph. But, current events have obviously had a huge impact on us all. It's difficult for me not to go on extensively about these issues, but I will restrain to this: we miss working together in our collaborations, we miss hosting and interacting in-person with our sponsors, we feel deeply for all who have suffered losses in their families and their livelihoods, and we remain dedicated to making PDL, CMU, and the world a place of equality, acceptance, and community. To all: be well, stay safe, and love thy neighbor!

Now, to the traditional function of this space: overviewing PDL's past year. It's always a pleasure to extol the successes of PDL's great people, and this year is no different. Some highlights include exciting new projects and collaborations with sponsors, continued growth and success for PDL's storage systems and cloud classes, and lots of great new activities and results in long-standing areas of strength like database systems, ML systems, and data processing infrastructure. Along the way, many students have graduated and joined PDL Consortium companies, PDL researchers have won some big awards, and many cool papers have been published. Specifics can be found throughout the newsletter, but let me highlight a few things.

Storage has been a PDL focus since its founding in the early 90s, and PDLers are exploring a number of new opportunities created by new storage technologies, new storage interfaces, and our connections to PDL Consortium companies. For example, building on our recent case study (see article) of Ceph evolution, we are exploring distributed storage back-ends specialized for zoned storage devices (e.g., ZNS SSDs or ZBR HDDs)... and we thank Western Digital for working with us to conduct real experiments. We are exploring new NVM-aware algorithms (including a recent Best Paper award) and NVM-specialized distributed redundancy approaches... and we thank Intel for enabling us to conduct real experiments. We are exploring Flash-specialized caching approaches, together with Facebook, and mechanical disk failure rates over time, with the help of NetApp and Google. The new insights and ideas arising from the ability to experiment with real devices, workload traces, and failure logs are exciting to see.

PDL has also long concerned itself with resource scheduling for data processing activities, and there are exciting new activities here as well, arising to address new computing models. For example, we have been working with Microsoft to study inter-job dependencies in data lakes and their impact on resource management. By analyzing provenance data and job logs from Cosmos, Microsoft's huge data lake infrastructure, we have been identifying significant challenges currently addressed in ad hoc ways and great opportunities for maximizing the value realized from a data analytics infrastructure (see article). Research on another front has devised new approaches for handling DNN training jobs in a shared cluster. By co-adaptively deciding per-job parameters (e.g., batch size) and how many nodes to use for each job, the aggregate performance of training jobs on the cluster can be improved significantly.

We continue to work on the challenges of high-performance and large-scale storage. Exciting new results have been achieved in our work on in-situ index creation for new data, including an R&D100 award for the DeltaFS work done jointly with LANL researchers. And, some surprising/cool applications for it

THE PDL PACKET

THE PARALLEL DATA LABORATORY

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

PUBLISHER

Greg Ganger

EDITOR

Joan Digney

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

THE PDL LOGO

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

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

PARALLEL DATA LABORATORY

FACULTY

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

George Amvrosiadis
David Andersen
Nathan Beckmann
Chuck Cranor
Lorrie Cranor
Christos Faloutsos
Saugata Ghose
Phil Gibbons
Garth Gibson
Mor Harchol-Balter
Gauri Joshi
Todd Mowry
David O'Hallaron
Andy Pavlo
Majd Sakr
M. Satyanarayanan
Rashmi Vinayak

STAFF MEMBERS

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

GRADUATE STUDENTS

Abutalib Aghayev
Daiyaan Arfeen
Mohammad Bakhshalipour
Nirav Atre
Ben Berg
Vilas Bhat
Amirali Boroumand
Sol Boucher
Matt Butrovich
Mengxin Cao
Dominic Chen
Zhiran Chen
Yae Jee Cho
Andrew Chung
Ziqi Dong
Pratik Fegade
Ziqiang Feng
Graham Gobieski
Zijing Gu
Samarth Gupta
Jin Han
Travis Hance
Ankush Jain
Angela Jiang
Ellango Jothimurugesan
Saurabh Arun Kadekodi
Daehyeok Kim
Thomas Kim
Arvind Sai Krishnan
Jack Kosaian
Joseph Koshakow
Michael Kuchnik
Anuva Kulkarni
Tian Li
Wan Shen Lim
Kaige Liu
Elliot Lockerman
Lin Ma
Ankur Mallick
Francisco Maturana
Sara McAllister
Charles McGuffey
Prashanth Menon
Hojin Park
Aurick Qiao
Brian Schwedock
Damla Senol
Baljit Singh
Vikramraj Sitpal
Suhas J Subramanya
Minh Truong
Dana Van Aken
Jianyu Wang
Ziqi Wang
Daniel Wong
Tong Xiao
Ricky Xu
Dongsheng Yang
Jason Yang
Zhengzhe Yang
Ling Zhang
Qing Zheng
Giulio Zhou

UNDERGRADUATE STUDENTS

Jordi Gonzalez
Julian Tutuncu-Macias

FROM THE DIRECTOR'S CHAIR

have been found (ask George, since I don't want to steal the story). Our HeART project continues, exploring how cluster storage redundancy can and should be adaptively specialized to the diverse observed failure rates of heterogeneous HDDs. Our new Pacemaker design regulates HeART activity to allow safe, efficient adaptive redundancy that can reduce by 20% of more the disks needed in 100K+ storage clusters... based on analyses of clusters at Google and Backblaze. There is also an exciting new project that explores how storage systems can be specialized for ML training workloads.

We are applying ML to data systems in many areas. Of course, Andy's effort to create self-driving database systems continues, with new active learning approaches, new auto-config approaches, and an emerging new database system. (He has over 20 students working on it!) We are also exploring ML-based admission policies for Flash caches, ML-based approaches for predicting when device failure rates will rise, and automated cloud storage provisioning given the huge set of available options.

Although I mentioned some examples above, there are other "systems for ML" projects ongoing. One of the coolest is the application of coded computation to ML inference to provide resilience to server failures (see front-page article). An emerging area of exploration focuses on simplifying MLops by exploiting provenance data and automating workflow exploration. Watch for cool results from this direction.

Many other ongoing PDL projects are also producing cool results... too many for me to cover. But, this newsletter and the PDL website offer more details and additional research highlights. Given the inability for us to gather in person, we did our first ever virtual PDL "Visit" Day. Attendance was strong (over 127 from our sponsor companies, compared to 35 at 2019's in-person PDL Visit Day), and we thank the many of you who participated. Based on the experience, we are planning to try some different ways of interacting with you remotely (stay tuned!), while looking forward to being able to celebrate an in-person PDL Retreat with you when circumstances permit!

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

2019 PDL Workshop and Retreat.

July 2020

- Aurick Qiao gave his speaking skills talk on "Pollux: Co-adaptive Cluster Scheduling for Goodput-Optimized Deep Learning."
- Andrew Chung presented his speaking skills talk on "Wing: Unearthing Inter-job Dependencies for Better Cluster Scheduling."

June 2020

- 22nd Annual/1st Virtual PDL Spring Visit Day.
- David O'Hallaron, Professor of ECE and CS awarded the Philip L. Dowd fellowship.
- Prashanth Menon delivered his thesis proposal "On Building Robustness into Compilation-Based Main-Memory Database Query Engines."
- Dana Van Aken delivered her thesis proposal "On Automatic Database Management System Tuning Using Machine Learning."
- Huanchen Zhang presented "Order-Preserving Key Compression for In-Memory Search Trees" at SIGMOD'20. Conglong Li presented "Improving Approximate Nearest Neighbor Search through Learned Adaptive Early Termination" and Lin Ma presented "Active Learning for ML Enhanced Database Systems" at the same conference.
- Ankur Mallick and co-authors received the best paper award at ACM SIGMETRICS 2020, for their paper on "Rateless Codes for Near-Perfect Load Balancing in Distributed Matrix-Vector Multiplication."
- Mor Harchol-Balter's team presented "Simple Near-Optimal Scheduling for the M/G/1" at SIGMETRICS.

Chuck Cranor, Qing Zheng and Ankush Jain enjoying the poster session at the 2019 PDL Retreat.

May 2020

- Rajat Kateja presented "TVARAK: Software-Managed Hardware Offload for Redundancy in Direct-Access NVM Storage" at the 47th Int'l Symposium on Computer Architecture.
- Phil Gibbons was a recipient of the 2019 Paris Kanellakis Theory and Practice Award.
- Mahadev Satyanarayanan was named a University Professor.
- Samarth Gupta gave a talk on "Correlated Multi-armed Bandits with a Latent Random Source" at ICASSP 2020. At the same conference, Jianyu Wang presented 2 papers: "Overlap Local-SGD: An Algorithmic Approach to Hide Communication Delays in Distributed SGD" and "Lookahead Converges to Stationary Points of Smooth Non-Convex Functions."
- The Allen Newell Award for Research Excellence winners include Lorrie Cranor, Lujo Bauer, and PDL Alumna Michelle Mazurek.
- Conglong Li defended his dissertaion on "Learned Adaptive Accuracy-Cost Optimization for Machine Learning Systems."

April 2020

- Jianyu Wang presented "SlowMo: Improving Communication-Efficient Distributed SGD with Slow Momentum" at ICLR 2020: The International Conference on Learning Representations.
- Angela Hao Jiang defended her dissertation on " Improving Deep Learning Training and Inference with Dynamic Hyperparameter Optimization."
- Sol Boucher proposed his PhD research on "Lightweight Preemptible Functions."
- Charles McGuffey proposed his PhD thesis topic on "Modernizing Models and Management of the Memory Hierarchy for Non-Volatile Memory."
- Rajat Kateja defended his dissertation on "Reducing Performance Overhead of Direct Access NVM Storage Redundancy."

March 2020

- Elliot Lockerman presented "Livia: Data-Centric Computing Throughout the Memory Hierarchy" at ASPLOS '20.
- "Writeback-Aware Caching," by Nathan Beckmann, Phillip B. Gibbons, Bernhard Haeupler, and Charles McGuffey won Best Paper at APOCS'20 in Salt Lake City, UT.
- Rashmi Vinayak won an NSF CAREER Award.
- Michael Kuchnik gave his speaking skills talk on "Progressive Compressed Records: Taking a Byte out of Deep Learning Data."

February 2020

- Daniel Berger co-authored "Learning Relaxed Belady for Content Distribution Network Caching" for NSDI '20 in Santa Clara, CA.
- Pratik Fegade gave a presentation on "Scalable Pointer Analysis of Data Structures using Semantic Models" at the 29th Conference on Compiler Construction (CC '20) in San Diego, CA, USA.
- Rashmi Vinayak won a Facebook Distributed Systems Research Award.
- Andrew Chung proposed his PhD research on "Realizing Value in Shared Compute Infrastructures."

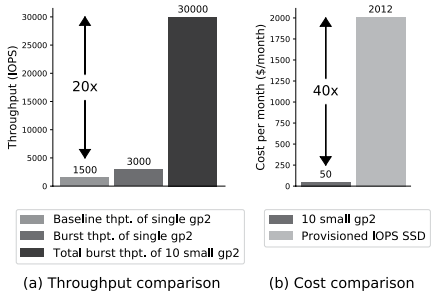
continued on page 33

More IOPS for Less: Exploiting Burstable Storage in Public Clouds

Hojin Park, Gregory R. Ganger, George Amvrosiadis

12th USENIX Workshop on Hot Topics in Cloud Computing (HotCloud '20). Virtual Boston, MA, July 13-14, 2020.

Burstable storage is a public cloud feature that enhances cloud storage volumes with credits that can be used to boost performance temporarily. These credits can be exchanged for increased storage throughput, for a short period of time, and are replenished over time. We examine how burstable storage can be leveraged to reduce cost and/or improve performance for three use cases with different data-longevity requirements: traditional persistent storage, caching, and ephemeral storage. Although cloud storage volumes are typically priced by capacity, we find that each AWS gp2 volume starts with the same number of burst credits. Exploiting that fact, we find that aggressive interchanging of large numbers of small short-term volumes can increase IOPS by up to 100 at a cost increase of only 10–40%. Compared to an AWS io1 volume provisioned for the same performance, such interchanging reduces cost by 97.5%.



Throughput comparison (left) of three storage configurations of the same cost. 10 small gp2 volumes provide 20× more IOPS (during burst) than a single volume. Purchasing this throughput directly increases cost by 40× (right).

DriftSurf: A Risk-competitive Learning Algorithm under Concept Drift

Ashraf Tahmasbi, Ellango Jothimurugesan, Srikanta Tirthapura, Phillip B. Gibbons

International Conference on Machine Learning (ICML) 2020. Virtual Vienna, Austria, July 12-18, 2020.

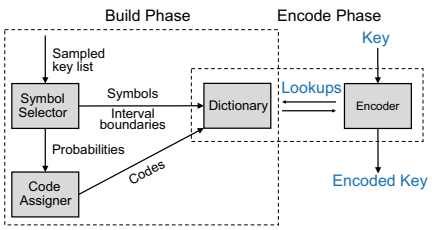
When learning from streaming data, a change in the data distribution, also known as concept drift, can render a previously-learned model inaccurate and require training a new model. We present an adaptive learning algorithm that extends previous drift-detection-based methods by incorporating drift detection into a broader stable-state/reactivestate process. The advantage of our approach is that we can use aggressive drift detection in the stable state to achieve a high detection rate, but mitigate the false positive rate of standalone drift detection via a reactive state that reacts quickly to true drifts while eliminating most false positives. The algorithm is generic in its base learner and can be applied across a variety of supervised learning problems. Our theoretical analysis shows that the risk of the algorithm is competitive to an algorithm with oracle knowledge of when (abrupt) drifts occur. Experiments on synthetic and real datasets with concept drifts confirm our theoretical analysis.

Order-Preserving Key Compression for In-Memory Search Trees

Huanchen Zhang, Xiaoxuan Liu, David G. Andersen, Michael Kaminsky, Kimberly Keeton, Andrew Pavlo

SIGMOD'20, June 14–19, 2020. Virtual Portland, OR.

We present the High-speed Order-Preserving Encoder (HOPE) for in-memory search trees. HOPE is a fast dictionary-based compressor that encodes arbitrary keys while preserving their order. HOPE's approach is to identify common key patterns at a fine granularity and exploit the entropy to achieve high compression rates with a small dictionary. We first develop a theoretical model to reason about order-preserving dictionary designs. We then select six representative compression schemes using this model and implement them in HOPE. These schemes make different trade-offs between compression rate and encoding speed. We evaluate HOPE on five data structures used in databases: SuRF, ART, HOT, B+tree, and Prefix B+tree. Our experiments show that using HOPE allows the search trees to achieve lower query latency (up to 40% lower) and better memory efficiency (up to 30% smaller) simultaneously for most string key workloads.



The HOPE Framework – An overview of HOPE's modules and their interactions with each other in the two phases.

TVARAK: Software-Managed Hardware Offload for Redundancy in Direct-Access NVM Storage

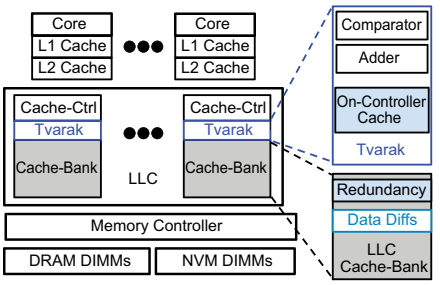
Rajat Kateja, Nathan Beckmann, Greg Ganger

47th International Symposium on Computer Architecture, May 30 – June 3, 2020, Virtual Valencia, Spain.

Production storage systems complement device-level ECC (which covers media errors) with system-checksums and cross-device parity. This system-level redundancy enables systems to detect and recover from data corruption due to device firmware bugs (e.g.,

continued on page 6

continued from page 5



TVARAK co-resides with the LLC bank controllers. It includes comparators to identify cache lines 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.

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. We propose to offload the update and verification of system-level redundancy to TVARAK, a new hardware controller colocated with the last-level cache. TVARAK enables efficient protection of data from such bugs in memory controller and NVM DIMM firmware. Simulation-based evaluation with seven data-intensive applications shows that TVARAK is efficient. For example, TVARAK reduces Redis set-only performance by only 3%, compared to 50% reduction for a state-of-the-art software-only approach.

Active Learning for ML Enhanced Database Systems

Lin Ma, Bailu Ding, Sudipto Das, Adith Swaminathan

SIGMOD'20, June 14–19, 2020. Virtual Portland, OR.

Recent research has shown promising results by using machine learning (ML) techniques to improve the performance of database systems, e.g., in query optimization or index recommendation. However, in many production deployments, the ML models' performance degrades significantly when the test data diverges from the data used to train these models.

In this paper, we address this performance degradation by using B-instances to collect additional data during deployment. We propose an active data collection platform, ADCP, that employs active learning (AL) to gather relevant data cost-effectively. We develop a novel AL technique, Holistic Active Learner (HAL), that robustly combines multiple noisy signals for data gathering in the context of database applications. HAL applies to various ML tasks, budget sizes, cost types, and budgeting interfaces for database applications. We evaluate ADCP on both industry-standard benchmarks and real customer workloads. Our evaluation shows that, compared with other baselines, our technique improves ML models' prediction performance by up to 2× with the same cost budget. In particular, on production workloads, our technique reduces the prediction error of ML models by 75% using about 100 additionally collected queries.

Machine Learning on Volatile Instances

Xiaoxi Zhang, Jianyu Wang, Gauri Joshi, Carlee Joe-Wong

IEEE Intl. Conf. on Computer Communications (INFOCOM). Virtual Toronto, Canada, July 6-9, 2020.

Due to the massive size of the neural network models and training datasets used in machine learning today, it is imperative to distribute stochastic gradient descent (SGD) by splitting up tasks such as gradient evaluation across multiple worker nodes. However, running distributed SGD can be prohibitively expensive because it may require specialized computing resources such as GPUs for extended periods of time. We propose cost-effective strategies that exploit volatile cloud instances that are cheaper than standard instances, but may be interrupted by higher priority workloads. To the best of our knowledge, this work is the first to quantify how variations in the number of active worker nodes (as a result of preemption) affects SGD convergence and the time to train the model. By understanding these tradeoffs between preemption probability of the instances, accuracy, and training time, we are able to derive practical strategies for configuring distributed SGD jobs on volatile instances such as Amazon EC2 spot instances and other preemptible cloud instances. Experimental results show that our strategies achieve good training performance at substantially lower cost.

Simple Near-Optimal Scheduling for the M/G/1

Ziv Scully, Mor Harchol-Balter, Alan Scheller-Wolf

Proceedings of the ACM Measurement and Analysis of Computer Systems - SIGMETRICS, June 2020, Boston, MA.

We consider the problem of preemptively scheduling jobs to minimize mean response time of an M/G/1 queue. When we know each job's size, the shortest remaining processing time (SRPT) policy is optimal. Unfortunately, in many settings we do not have access to each job's size. Instead, we know only the job size distribution. In this setting the Gittins policy is known to minimize mean response time, but its complex priority structure can be computationally intractable. A much simpler alternative to Gittins is the shortest expected remaining processing time (SERPT) policy. While SERPT is a natural extension of SRPT to unknown job sizes, it is unknown whether or not SERPT is close to optimal for mean response time.

We present a new variant of SERPT called monotonic SERPT (M-SERPT) which is as simple as SERPT but has provably near-optimal mean response time at all loads for any job size distri-

continued on page 7

continued from page 6

bution. Specifically, we prove the mean response time ratio between M-SERPT and Gittins is at most 3 for load $\rho \leq 8/9$ and at most 5 for any load. This makes M-SERPT the only non-Gittins scheduling policy known to have a constant-factor approximation ratio for mean response time.

Improving Approximate Nearest Neighbor Search through Learned Adaptive Early Termination

Conglong Li, Minjia Zhang, David G. Andersen, Yuxiong He

SIGMOD '20, June 14–19, 2020, Virtual Portland, OR, USA.

In applications ranging from image search to recommendation systems, the problem of identifying a set of "similar" real-valued vectors to a query vector plays a critical role. However, retrieving these vectors and computing the corresponding similarity scores from a large database is computationally challenging. Approximate nearest neighbor (ANN) search relaxes the guarantee of exactness for efficiency by vector compression and/or by only searching a subset of database vectors for each query. Searching a larger subset increases both accuracy and latency. State-of-the-art ANN approaches use fixed configurations that apply the same termination condition (the size of subset to search) for all queries, which leads to undesirably high latency when trying to achieve the last few percents of accuracy. We find that due to the index structures and the vector distributions, the number of database vectors that must be searched to find the ground-truth nearest neighbor varies widely among queries. Critically, we further identify that the intermediate search result after a certain amount of search is an important runtime feature that indicates how much more search should be performed.

To achieve a better tradeoff between latency and accuracy, we propose a novel approach that adaptively determines search termination conditions for individual queries. To do so, we build and train gradient boosting decision tree models to learn and predict when to stop searching for a certain query. These models enable us to achieve the same accuracy with less total amount of search compared to the fixed configurations. We apply the learned adaptive early termination to state-of-the-art ANN approaches, and evaluate the end-to-end performance on three million to billion-scale datasets. Compared with fixed configurations, our approach consistently improves the average end-to-end latency by up to 7.1 times faster under the same high accuracy targets. Our approach is open source at github.com/efficient/faisslearned-termination.

Overlap Local-SGD: An Algorithmic Approach to Hide Communication Delays in Distributed SGD

Jianyu Wang, Hao Liang, Gauri Joshi

International Conference on Acoustics, Speech, and Signal Processing (ICASSP) 2020. Virtual Barcelona, Spain, May 4-8, 2020.

Distributed stochastic gradient descent (SGD) is essential for scaling the machine learning algorithms to a large number of computing nodes. However, the infrastructures variability such as high communication delay or random node slowdown greatly impedes the performance of distributed SGD algorithm, especially in a wireless system or sensor networks. In this paper, we propose an algorithmic approach named Overlap-Local-SGD (and its momentum variant) to overlap the communication and computation so as to speedup the distributed training procedure. The approach can help to mitigate the straggler effects as well. We achieve this by adding an anchor model on each node. After multiple local updates, locally trained models will be pulled back towards the synchronized anchor model rather than communicating with others. Experimental results of training a deep neural network on CIFAR-10 dataset demonstrate the effectiveness of Overlap-Local-SGD. We also provide a convergence guarantee for the proposed algorithm under non-convex objective functions.

Lookahead Converges to Stationary Points of Smooth Non-Convex Functions

Jianyu Wang, Vinayak Tantia, Nicolas Ballas, Michael Rabbat

ICASSP 2020: 45th International Conference on Acoustics, Speech, and Signal Processing. Virtual Barcelona, Spain, May 4-8, 2020.

The Lookahead optimizer [Zhang et al., 2019] was recently proposed and demonstrated to improve performance of stochastic firstorder methods for training deep neural networks. Lookahead can be viewed as a two time-scale algorithm, where the fast dynamics (inner optimizer) determine a search direction and the slow dynamics (outer optimizer) perform updates by moving along this direction. We prove that, with appropriate choice of step-sizes, Lookahead converges to a stationary point of smooth non-convex functions. Although Lookahead is described and implemented as a serial algorithm, our analysis is based on viewing Lookahead as a multi-agent optimization method with two agents communicating periodically.

Correlated Multi-armed Bandits with a Latent Random Source

Samarth Gupta, Gauri Joshi, Osman Yagan

International Conference on Acoustics, Speech, and Signal Processing (ICASSP) 2020. Virtual Barcelona, Spain, May 4-8, 2020.

We consider a novel multi-armed bandit framework where the rewards

continued on page 23

AWARDS & OTHER PDL NEWS

July 2020
David O'Hallaron Awarded the Philip L. Dowd Fellowship

Congratulations to ECE and CS Professor David O'Hallaron, who has been awarded the Philip L Dowd Fellowship in the College of Engineering. The fellowship is awarded to recognize educational contributions and encourage the undertaking of an educational project such as textbook writing, educational technology development, laboratory experience improvement, educational software, or course and curriculum development. The Dowd Fellowship Award usually consists of a memento and a discretionary fund to support the nominee's education project and lasts for one year beginning the January following the award.

July 2020
A New PDL Grandchild!

Welcome to Olivia Mae Losi, Karen Lindenfelser's fourth grandchild! Olivia was welcomed by Laura and Pete and sisters Layla and Nora, on July 23, 2020. She arrived at 11:38 am; 6 lbs. 13 ounces and 20 inches, a bundle of indescribable joy!

June 2020
Best Paper at SIGMETRICS'20!

Congratulations to Ankur Mallick, Malhar Chaudhari, Ganesh Palanikumar, Utsav Sheth, and Gauri Joshi on receiving the best paper award at the Association for Computing Machinery's (ACM) annual SIGMETRICS conference, which was held virtually in Boston, MA, June 8-12. Their paper, "Rateless Codes for Near-Perfect Load Balancing in Distributed Matrix-Vector Multiplication," proposes a rateless fountain coding strategy that its latency is asymptotically equal to ideal load balancing, and it performs asymptotically zero redundant computation.

May 2020
Phil Gibbons Named Recipient of 2019 Paris Kanellakis Theory and Practice Award

The Paris Kanellakis Theory and Practice Award honors specific theoretical accomplishments that have had a significant and demonstrable effect on the practice of computing. This award is accompanied by a prize of $10,000 and is endowed by contributions from the Kanellakis family, with additional financial support provided by ACM's Special Interest Groups on Algorithms and Computational Theory (SIGACT), Design Automaton (SIGDA), Management of Data (SIGMOD), and Programming Languages (SIGPLAN), the ACM SIG Projects Fund, and individual contributions.

ACM has named Noga Alon of Princeton University and Tel Aviv University; Phillip Gibbons of Carnegie Mellon University; Yossi Matias of Google and Tel Aviv University; and Mario Szegedy of Rutgers University recipients of the ACM Paris Kanellakis Theory and Practice Award for seminal work on the foundations of streaming algorithms and their application to large-scale data analytics.

Alon, Gibbons, Matias and Szegedy pioneered a framework for algorithmic treatment of streaming massive datasets. Today, their sketching and streaming algorithms remain the core approach for streaming big data and constitute an entire subarea of the field of algorithms. Additionally, the concepts of sketches and synopses that they introduced are now routinely used in a variety of data analysis tasks in databases, network monitoring, usage analytics in internet products, natural language processing and machine learning.

In their seminal paper, "The Space Complexity of Approximating the Frequency Moments," Alon, Matias and Szegedy laid the foundations of the analysis of data streams using limited memory. Follow-up papers, including "Tracking Join and Self-join Sizes in Limited Storage," by Alon, Gibbons, Matias, and Szegedy, and "New Sampling-Based Summary Statistics for Improving Approximate Query Answers," by Gibbons and Matias, expanded on the idea of data synopses and were instrumental n the development of the burgeoning fields of streaming and sketching algorithms. This work has been applied to query planning and processing in databases and the design of small synopses to monitor vast quantities of data generated in networks.

-- acm.org news, May 2020

May 2020
M. Satyanarayanan Named A University Professor

Seven Carnegie Mellon University faculty members have been elevated to the rank of University Professor, the highest distinction a faculty member can achieve at CMU. Among the newly

continued on page 9

continued from page 8

appointed University Professors is Mahadev Satyanarayanan, a faculty member of the PDL.

"University Professors are distinguished by international recognition and for their contributions to education, artistic creativity and/or research," said Provost Jim Garrett. "Each University Professor exemplifies a high level of professional achievement and an exceptional commitment to academic excellence at our university." Garrett said professors are nominated and recommended by academic leaders and faculty who have already achieved the designation of University Professor.

Mahadev Satyanarayanan, best known as Satya, is the Carnegie Group Professor of Computer Science. He is an experimental computer scientist, who designs, implements and evaluates systems. Satya's multi-decade research career has focused on the challenges of performance, scalability, availability and trust in information systems that reach from the cloud to the mobile edge of the Internet.

In the course of this work, he has pioneered many advances in distributed systems, mobile computing, pervasive computing, and the Internet of Things (IoT). Examples include his work as principal architect of CMU's pioneering Andrew File System in the 1980s, his work on disconnected and weakly connected mobile data access in the Coda File System in the early 1990s, his work on adaptive and energy-efficient mobile computing in the Odyssey system in the late 1990s and early 2000s and his most recent work in establishing the field of edge computing.

Satya earned his Ph.D. in computer science at CMU in 1983. An Association for Computing Machinery (ACM) and IEEE Fellow, he is the recipient of numerous awards, including the Association for Computing Machinery's prestigious Software System Award for his work on the Andrew File System.

-- with info from CMU News, May 15, 2020

May 2020
Congratulations to 2020 SCS Faculty Award Winners!

Congratulations to these Allen Newell Award for Research Excellence winners: Lujo Bauer, Nicolas Christin, Lorrie Cranor, Saranga Komanduri, Michelle Mazurek, William Melicher, Sean Segretti, Rich Shay and Blase Ur, for their pioneering contribution to the science of evaluating password strength and for embodying this science in online tools that enable individuals and groups to more easily secure their systems.

-- Cylab News, May 22, 2020

March 2020
Rashmi Vinayak Wins NSF CAREER Award

Rashmi Vinayak, an assistant professor in the Carnegie Mellon University Computer Science Department, has won a five-year, $650,000 Faculty Early Career Development (CAREER) Award, the National Science Foundation's most prestigious award for young faculty members.

The award will support Vinayak's work to improve the resource and energy efficiency of large-scale data centers, which together serve as the backbone for internet-based services, cloud services and data analytics platforms.

"Such large-scale systems are prone to failures and unavailability, and therefore have a high degree of redundancy built in to them to provide resilience against such events," she noted. "While redundancy provides resilience, it comes with a significant overhead in terms of resource and energy requirements. The overarching goal of this project is to design resource- and energy-efficient redundancy algorithms for data centers using tools based on information theory and coding theory."

Vinayak earned her Ph.D. in electrical engineering and computer science at the University of California at Berkeley, where she also worked as a postdoctoral researcher before joining CSD in 2017. Her previous awards include the Eli Jury Award from Berkeley's EECS Department, a Google Faculty Research Award, Facebook Communications and Networking Research Award, and the Tata Institute of Fundamental Research Memorial Lecture Award.

-- Carnegie Mellon University News, March 20, 2020

March 2020
APOCS'20 Best Paper!

The inaugural APOCS (SIAM Symposium on Algorithmic Principles of Computer Systems), held in January in Salt Lake City, UT, awarded its best paper accolades to a CMU team for

continued on page 10

Chad Dougherty attending to the equipment in the DCO during the quarantine.

continued from page 9

the paper "Writeback-Aware Caching", by Nathan Beckmann, Phillip B. Gibbons, Bernhard Haeupler, and Charles McGuffey. The paper explores the Writeback-Aware Caching problem, which modifies traditional caching problems by explicitly accounting for the cost of writing modified data back to memory on eviction. Congratulations!

February 2020
Rashmi Vinayak Wins Facebook Distributed Systems Research Award

Congratulations to Rashmi on receiving a Facebook Distributed Systems Research Award for her work on "Reduced cost cluster storage by exploiting disk-reliability heterogeneity."

Facebook launched the Distributed Systems Award at the Symposium on Operating Systems Principles in October 2019 to foster forward-looking research in the area of distributed systems, applying important techniques from the field at Facebook's scale and sharing our designs, implementations, insights, and data with the community.

Out of a total of 63 proposals from 12 countries and 50 universities, eight winners were selected and are listed below. "We were thrilled to receive so many high-quality and thought-provoking submissions; we continue to be inspired by the work of our academic peers," says PDL Alumni Justin Meza, Research Scientist on the Facebook Core Systems team.

"It was challenging to only select eight awardees," he says. "We are grateful to the research community for engaging so enthusiastically with us, and we look forward to our continued collaboration." The RFP winners are invited to the Core Systems Faculty Summit in 2020 (time TBD), where they will have the opportunity to discuss their proposals with the research community.

--with info from Facebook Research News, Feb. 2020

February 2020
Welcome Isaac

Huanchen Zhang and his wife would like us to meet their first child, Isaac Zhang who was born February 26, 2020, at 3:28pm, weighing 6lb 8oz, and measuring 18.5'' long. Mom and baby are healthy, He has already melted their hearts! Life is a miracle!

January 2020
Juncheng (Jason) Yang Receives Facebook Fellowship

Four Ph.D. candidates in the School of Computer Science are among 36 outstanding students in computer science and engineering from 16 universities who have been named 2020 recipients of the Facebook Fellowship Program.

Each Facebook fellow receives tuition and fees for up to two academic years and a stipend of $42,000, which includes conference travel support. Facebook received applications from 1,876 students at more than 100 universities for this year's program.

Among the four CMU students receiving the award is Juncheng Yang, a Ph.D. student in the Computer Science Department, to be a fellow in computer storage and efficiency. Yang is broadly interested in the reliability, performance and availability in the storage and caching subsystems of internet-scale web services.

--CMU School of Computer Science News, Jan. 29, 2020

January 2020
Rashmi Vinayak Receives Prof. R. Narasimhan Memorial Lecture Award

Assistant Professor Rashmi Vinayak received the "Prof. R. Narasimhan memorial lecture award 2020" from Tata Institute of Fundamental Research (TIFR), which is a premier research institution in India. She delivered the memorial lecture at TIFR on January 6th 2020 concerning her work on "Convertible codes: New class of codes for efficient conversion of coded data in distributed storage."

She leads the CMU TheSys research group, which is a part of the Parallel Data Lab (PDL). Rashmi's research interests lie in the broad area of computer and networked systems with a current focus on reliability, availability, scalability, and performance challenges in data storage and caching systems, in systems for machine learning and in live video streaming.

January 2020
Mor Harchol-Balter Awarded Bruce J. Nelson Chair in Computer Science

Congratulations to Mor, who was awarded the Dr. Bruce J. Nelson Professorship in Computer Science.

The chair was created by the family and friends of Bruce Nelson (CS'81) in 2000, in his memory, to support a faculty member in the School of Computer Science.

October 2019
DeltaFS Project Takes Home R&D 100 Award

The R&D World Magazine announced its 100 winners for 2019

continued on page 11

continued from page 10

on October 29. "These 100 winning products and technologies are the disruptors that will change industries and make the world a better place in the coming years," said Paul J. Heney, Vice President, Editorial Director for R&D World.

In the IT/Electrical category, the winner is the CMU/Los Alamos National Laboratory collaborative project "DeltaFS—Rapidly Searching Big Data." Congratulations to its may contributors, including Brad Settlemyer, Scientist, Los Alamos National Laboratory; George Amvrosiadis, Research Professor Carnegie Mellon University; Gary Grider, HPC Division Director, Los Alamos National Laboratory; Qing Zheng, Research Assistant, Carnegie Mellon University; Greg Ganger, Jatras Professor, Carnegie Mellon University; Charles Cranor, Systems Scientist, Carnegie Mellon University; and Garth Gibson, Professor, Carnegie Mellon University

--RDWorld Online, October 29, 2019

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.

October 2019
Welcome Clara!

```
INSERT INTO people
  (name, sex, dob,
   mother, alleged_father)
VALUES
  ('Clara X Pavlo', 'F', '2019-10-24',
   @DeepGenes, @andy_pavlo);
```

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 Lab.

-- SCS News - September 12, 2019

April 2019
Schwedock receives NSF Graduate Research Fellowship

Brian Schwedock, an electrical and 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.

continued on page 12

AWARDS & OTHER PDL NEWS

continued from page 11

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

In Memoriam
David B. Anderson

We were deeply saddened by the passing in October 2019 of one of PDL's foremost industry supporters. Dave Anderson's career with Seagate Technology in Shakopee, MN as Director of Strategic Planning and as Technologist spanned almost 40 years and he participated in the PDL Retreats from almost the beginning. He is remembered by all in the PDL for his insight and encouragement of our research. For many of us, he will also be missed at other industry/university connection-points (FAST leadership meetings, as guest lecturer in our storage systems class, etc.). Dave's unique combination of a dry sense of humor and breadth of knowledge, both inside and outside our industry, are very much missed, but we are all better for having known him.

BETTER SCHEDULING IN DATA LAKES

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

Andrew Chung, Subru Krishnan*, Konstantinos Karanasos*, Carlo Curino*, Greg Ganger; *Microsoft

Shared data analytics infrastructures or "data lakes" have become core elements of modern data-driven enterprises, providing required data storage and analysis infrastructure (see Fig. 1). Data lakes enhance data processing via a combination of two critical properties: (i) a highly consolidated, multi-tenant infrastructure that enables multiple teams of data scientists and engineers to share resources, and (ii) easy data sharing between users and various types of data analytics applications. These properties increase data re-use and reduce overall computational resource-hours consumed.

This data and resource sharing also creates a new challenge: hidden inter-job dependencies. We say that Job B depends on Job A if Job B takes as input any output file generated and stored into a shared distributed file system by Job A. If Job C depends on Job B, it follows that it also has an (indirect) dependency on Job A. We refer to these as hidden dependencies, in contrast with explicit computation DAGs managed by schedulers within workflow managers or DBMSs, because no indication of such dependencies is available in job submissions.

Tracking data provenance and data movement both within the data lake and external components create an unprecedented opportunity to uncover and exploit these inter-job dependencies. As a case study, we analyzed petabytes of job and data provenance logs for 90 days of a 50k+ server cluster at Microsoft shared by over 1300 users from more than 150 internal organizations covering over

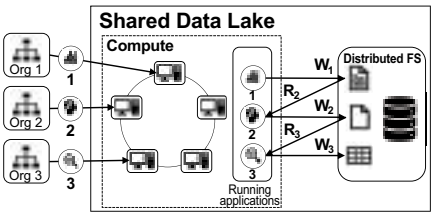


Figure 1: Data lake overview. Different types of applications, submitted by different organizations, share the same compute infrastructure and read (R) and write (W) to the same storage system, thereby creating inter-job dependencies as jobs consume the output of other jobs. For example, Job 2 (from Org 2) reads a file written by Job 1, so Job 2 depends on Job 1.

4 million submitted jobs and their 16 million inter-job dependencies. Almost 80% of the submitted jobs depended on output generated by at least one other

continued on page 13

continued from page 12

job. Many of the dependencies were shown as cross-organization, with 20% of jobs depending on jobs submitted by another organization.

Despite so much inter-job dependence, systems provide little support for addressing associated challenges. For example, even in the expertly-managed data lake we studied, different users and organizations make their own decisions regarding when to submit jobs and how to set job priorities. We found that 34% of recurring jobs played a submission-timing game, where they are submitted without checking if job inputs they depend on are available, failing immediately if they are not.

In response, we have developed the Wing dependency profiler, which efficiently processes historical job and provenance data to predict the impact of each new job on future jobs and user downloads. Although it is inherently difficult to know what future jobs will depend on the output generated by a current job, Wing finds success by focusing on recurrence.

Previous workload studies have shown that most jobs in data analytics environments are recurrent. Our analysis finds that inter-job dependency patterns are similarly recurrent, with jobs of the same template following similar input dependency patterns. Therefore, Wing is able to use historically recurring dependencies to (i) analyze and predict relationships between common, dependent recurring jobs, and (ii) guide a cluster scheduler to value jobs in a way that accounts for hidden dependencies.

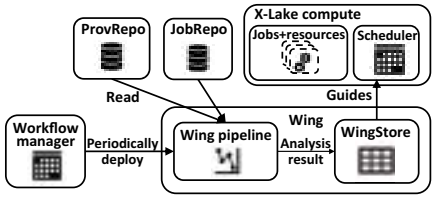


Figure 2: Wing architecture. The workflow manager periodically deploys Wing's analysis pipeline to X-Lake. Upon pipeline completion, results of its analyses are loaded in to WingStore, which informs Wing-guided schedulers with job and dependency characteristics.

Pairing Wing with traditional YARN scheduling replaces user-provided priorities with what Wing believes is a job's aggregate value. Experiments showed that inter-job dependency-aware guidance from Wing enables attainment of up to 66% more value compared to the default priority-based scheduling policy used in our data lake under cluster capacity crunch. We also found that when organizational cluster resource boundaries are removed, a Wing-guided scheduler can attain up to 93% of total value while using only 20% of total cluster capacity. In simulations driven by real job logs, a traditional YARN scheduler that uses Wing-provided values in place of user-specified priorities extracts more value (in terms of successful dependent jobs and user downloads) from a heavily-loaded cluster. By relying completely on Wing for guidance, YARN can achieve nearly 100% of the value with 80% fewer machines, a 4X improvement over user-provided priorities and organizational queues.

Companies are always seeking to make the most out of infrastructure investments. Compute-hungry and opportunistic applications such as parameter exploration for machine learning are becoming more popular. This creates the opportunity and need for more sophisticated schedulers that can identify which jobs are the most consequential. Similarly, as operators become more efficient in operating clusters, unexpected global events (such as COVID-19) push infrastructures in to uncharted territories. Evaluation of job importance becomes key, and we find that guidance from Wing can help schedulers achieve higher value in the face of increasing amounts of resource demand, without requiring the deployment of significantly more compute resources.

THE CASE FOR CUSTOM STORAGE BACKENDS IN DISTRIBUTED STORAGE SYSTEMS

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

Distributed file systems aggregate storage space from multiple physical machines into a single unified data store that offers high-bandwidth and parallel I/O, horizontal scalability, fault tolerance, and strong consistency. While distributed file systems may be designed differently and use unique terms to refer to the machines managing data placement on physical media, the storage backend is usually defined as the software module directly managing the storage device attached to physical machines. In these systems, the storage backend is the software module that manages space on disks (OSTs, Bricks, OSDs) attached to physical machines (OSSs, Nodes).

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. Three recurring challenges arose. First, developing a zero-overhead transaction mechanism is challenging. Second, metadata performance at the local level can signifi-

continued on page 14

continued from page 13

cantly affect performance at the distributed level. Third, supporting emerging storage hardware is painstakingly slow.

BLUESTORE

In 2015, the Ceph project started designing and implementing BlueStore, a user space storage backend that stores data directly on raw storage devices and metadata in a key-value store. By taking full control of the I/O path, BlueStore has been able to efficiently implement full data checksums, inline compression, and fast overwrites of erasure-coded data, while also improving performance on common customer workloads. With this control, for example, BlueStore can choose the checksum block size based on the I/O hints, and if the hints indicate that objects are to be compressed, then a checksum can be computed after the compression, significantly reducing the total size of checksum metadata.

To achieve fast metadata operations, it stores metadata in RocksDB. To avoid consistency overhead, BlueStore writes data directly to raw disk, resulting in one cache flush for data write. Then, RocksDB is updated to reuse WAL files as a circular buffer, resulting in one cache flush for metadata write. For incoming writes larger than a minimum allocation size (64 KiB for HDDs, 16 KiB for SSDs), the data are written to a newly allocated extent.

BlueStore's full I/O control also allows it to provide an efficient clone operation. A clone operation simply increments the reference count of dependent extents, and writes are directed to new extents. The same primitive also allows BlueStore to avoid journal double writes for object writes and partial overwrites that are larger than the minimum allocation size. For writes smaller than the minimum allocation size, both data and metadata are first inserted to RocksDB as promises of future I/O, and then asynchronously written to disk after the transaction commits. This deferred write mechanism has two purposes. First, it batches small writes to increase efficiency, because new data writes require two I/O operations whereas an insert to RocksDB requires one. Second, it optimizes I/O based on the device type; 64-KiB (or smaller) overwrites of a large object on an HDD are performed asynchronously in place to avoid seeks during reads, whereas in-place overwrites only happen for I/O sizes less than 16 KiB on SSDs.

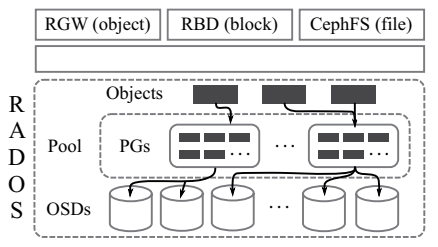


Fig. 1. High-level depiction of Ceph's architecture. A single pool with 3× replication is shown. Therefore, each placement group (PG) is replicated on three OSDs.

Supporting Shingled Magnetic Recording (SMR) hard disks, which increase drive capacities by partially overlapping adjacent magnetic tracks, is also important for scale-out distributed file systems, because it lowers storage costs. Since BlueStore offers us the freedom to explore novel interfaces and data layouts, we looked toward adopting HM-SMR drives with the new zone interface and as a first step, we adapt RocksDB, an LSM-Tree instance, to run on zoned drives [2].

SUMMARY

Our experience with Ceph shows the belief that developing a mature, production-ready storage backend from scratch is a time-consuming process is inaccurate. Furthermore, we find that developing a special-purpose, user space storage backend from scratch (1) reclaims the significant performance left on the table when building a backend on a general-purpose file system, (2) makes it possible to adopt novel, backward incompatible storage hardware, and (3) enables new features by gaining complete control of the I/O stack.

Jason Boles keeps all the technical aspects of the 2019 PDL Retreat running smoothly.

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 backward-incompatible storage hardware possible, an important trait in a changing storage landscape that is learning to embrace hardware diversity. For more information, please see the SOSP 2019 paper [1].

REFERENCES

[1] File Systems Unfit as Distributed Storage Backends: Lessons from 10 Years of Ceph Evolution. A. Aghayev, S. Weil, M. Kuchnik, M. Nelson, G.R. Ganger, G. Amvrosiadis. SOSP '19, October 27–30, 2019, Huntsville, ON, Canada. https://www.pdl.cmu.edu/PDL-FTP/Storage/ceph-exp-sosp19_abs.shtml

[2] Reconciling LSM-Trees with Modern Hard Drives using BlueFS. A. Aghayev, S. Weil, G.R. Ganger, G. Amvrosiadis. Carnegie Mellon University Parallel Data Lab Technical Report CMU-PDL-19-102, April 2019. https://www.pdl.cmu.edu/PDL-FTP/FS/CMU-PDL-19-102_abs.shtml

continued from page 1

Erasure codes are popular tools for imparting resilience to data unavailability while remaining agnostic to the cause of unavailability and using less resources than replication-based approaches. These properties have led to the wide adoption of erasure codes in storage and communication systems. An erasure code encodes k data units to produce r redundant "parity" units in such a way that any k of the total (k + r) data and parity units are sufficient for a decoder to recover the original k data units. The overhead incurred by an erasure code is ((k+r) / k), which is typically much less than that of replication.

A number of recent works have studied the theoretical aspects of using erasure codes for alleviating the effects of slowdowns and failures that occur in distributed computation. This setup, called "coded computation," uses erasure coding to recover the outputs of a deployed computation over data units. In coded computation, data units are encoded into parity units, and the deployed computation is performed over all data and parity units in parallel. A decoder then uses the outputs from the fastest k of these computations to reconstruct the outputs corresponding to the original data units. For a prediction-serving system, employing coded computation would involve encoding queries such that a decoder can recover slow or failed predictions.

The primary differences between coded computation and the traditional use of erasure codes in storage and communication come from (1) performing computation over encoded data and (2) the need for an erasure code to recover the results of computation over data units rather than the data units themselves. Whereas traditional applications of erasure codes involve encoding data units and decoding from a subset of data and parity units, in coded computation one decodes by using the output of computation over data and parity units. This difference calls for fundamentally rethinking the design of erasure codes, as many of the erasure codes which have been widely used in storage and communication are applicable only to a highly restricted class of computations.

As erasure codes can correct slowdowns with low latency and require less resource-overhead than replication-based approaches, enabling the use of coded computation in prediction-serving systems has potential for efficient mitigation of tail latency inflation. However, the complex non-linear components common to popular machine learning models, such as neural networks, make it challenging to design effective coded-computation solutions for prediction-serving systems. Existing coded-computation approaches, which focus on hand-crafting new erasure codes, can support only rudimentary computations, rendering them inadequate for prediction-serving systems.

We propose to overcome the challenges of employing coded computation for prediction-serving systems via a learning-based approach. We show that machine learning can eschew the difficulty of hand-crafting codes and enable coded computation over neural network inference. However, this approach requires careful consideration: we show that simply replacing encoders and decoders with machine learning models limits opportunities to reduce tail latency.

Motivated by these insights, we take a fundamentally new approach to coded computation by introducing new components that we call "parity models." A parity model is a neural network trained to convert encoded queries into a form that enables decoding of unavailable predictions. Unlike conventional coded-computation approaches, which design new erasure codes, parity models enable the use of simple, fast encoders and decoders, such as addition and subtraction. Compared to learning encoders and decoders, this new approach reduces the computational burden introduced on a prediction-serving system's frontend, and also reduces the latency of reconstructions. We implement parity models in ParM, a prediction-serving system designed to make use of erasure-coded resilience. ParM encodes multiple queries together into a parity query, and a deployed parity model transforms the parity query such that its output enables a decoder to reconstruct slow or failed predictions.

The predictions returned by ParM are the same as those returned by any prediction-serving system in the absence of slowdowns and failures. When slowdowns and failures do occur, the output of ParM's decoder is an approximate reconstruction of a slow or failed prediction. Reconstructing approximations of unavailable predictions is appropriate for inference, as predictions themselves are approximate, and because ParM's reconstructions are returned only when a prediction would otherwise be slow or failed. In this scenario, it is often preferable to return an approximate prediction rather than a late one.

We evaluate the accuracy of ParM's reconstructions on a variety of neural networks and inference tasks such as image classification, speech recognition, and object localization. We also evaluate ParM's ability to reduce tail latency in the presence of resource contention. ParM reconstructs unavailable predictions with high accuracy and reduces tail latency while using 2-4× less additional resources than replication-based approaches. For example, using only half of the additional resources as replication, ParM's reconstructions from ResNet-18 models on various tasks are within a 6.5% difference in accuracy compared to if the original predictions were not slow or failed. Furthermore, ParM brings tail latency up to 3.5× closer to median latency, while maintaining the same median. This enables ParM to maintain predictable latencies in the face of slowdowns and failures. These results show the promise of learning-based coded computation to open new doors for imparting efficient resilience to prediction-serving systems.

DISSERTATION ABSTRACT: Learned Adaptive Accuracy-Cost Optimization for Machine Learning Systems

Conglong Li
Carnegie Mellon University, SCS

PhD Defense — May 1, 2020

This dissertation seeks to address the challenge of making adaptive accuracy-cost balancing inside systems for large-scale machine learning-based recommendation services. We show that it is important to make performance tradeoff decisions at a per-query basis instead of a predefined policy for all queries. We show that we can achieve a better tradeoff between accuracy and cost by leveraging lightweight machine learning models to make more adaptive decision-making inside systems infrastructure.

Large-scale recommendation services have two computation-heavy components with strict accuracy and latency targets: scoring (typically achieved by complex machine learning models) and candidate retrieval (typically achieved by approximate nearest neighbor search). We first introduce a caching system for scoring component in recommendation systems (in particular search advertising systems). Inside the cache, we leverage lightweight machine learning models to make adaptive cache refresh decisions, which provides a better balance between recommendation accuracy and computation cost. This leads to a better net profit in the search advertising context. We then present the learned adaptive termination for approximate nearest neighbor search inside the candidate retrieval component. We leverage lightweight machine learning models to decide how much to search for each query, which provides a better balance between the search accuracy and latency (computation cost).

Rajat Kateja at his desk at home after delivering his dissertation defense remotely in May.

DISSERTATION ABSTRACT: Improving Deep Learning Training and Inference with Dynamic Hyperparameter Optimization

Angela Hao Jiang
Carnegie Mellon University, SCS

PhD Defense — April 29, 2020

Over the past decade, deep learning has demonstrated high accuracy on challenges in fields like computer vision and natural language processing, revolutionizing these fields in the process. Deep learning models are now a fundamental building block for applications such as autonomous driving, medical imaging, and machine translation. However, many challenges remain when deploying these models in production. Researchers and practitioners must address a diversity of questions, including how to efficiently design, train, and deploy resource-intensive deep learning models and how to automate these approaches while ensuring robustness to changing conditions.

Our work aims to improve the efficiency of deep learning training and inference, as well as the underlying systems' robustness to changes in the environment. We address these issues by focusing on the many hyperparameters that are tuned to optimize the model's accuracy and resource usage. These hyperparameters include the choice of model architecture, the training dataset, the optimization algorithm, the hyperparameters of the optimization algorithm (e.g., the learning rate and momentum) and the training time budget. Currently, in practice, almost all hyperparameters are tuned once before training and held static. This is sub-optimal as the conditions that dictate the best hyperparameter value change over time (e.g., training progress, inference hardware). We apply dynamic tuning to hyperparameters that have traditionally been considered static. Using three case studies, we show that using runtime information to dynamically adapt hyperparameters that are traditionally static, such as the emphasis on individual training examples and the weights updated during transfer learning, can increase the efficiency of deep learning training and inference.

DISSERTATION ABSTRACT: Reducing Performance Overhead of Direct Access NVM Storage Redundancy

Rajat Kateja
Carnegie Mellon University, ECE

PhD Defense — April 10, 2020

Non-volatile memory (NVM) based storage is poised for mainstream deployment. DIMM form-factor NVM devices reside on the memory bus and offer DRAM-like access granularities and latencies along with non-volatility. NVM's Direct Access (DAX) interface enables applications to map persistent data into their address space and access it with load and store instructions, eliminating system software overheads.

Production deployment of DAX NVM storage would require that the storage system offer resilience against firmware-bug-induced data corruption, akin to conventional storage systems. Protection against firmware-bug-induced data corruptions requires

continued on page 17

continued from page 16

the storage system to maintain system-level redundancy, which we refer to as system-redundancy. With DAX interfacing, the lack of interposed system software makes it challenging to identify data reads and writes that should trigger system-redundancy verification and updates, respectively. Further, the DAX granularities (e.g., 64-byte cache lines) are incongruent with typical system-redundancy granularities (e.g., 4K pages), leading to high performance overhead in maintaining system-redundancy.

This dissertation demonstrates that DAX NVM storage systems can efficiently maintain system-redundancy by relaxing the data coverage guarantees or by leveraging a hardware offload. We support the thesis with two case studies: Vilamb and Tvarak.

The Vilamb library maintains system-redundancy asynchronously, avoiding critical path interpositioning and amortizes the overhead of system redundancy updates across multiple writes to a page. As a result, Vilamb provides 3 -5× the throughput of the state-of-the-art software solution at high operation rates. For applications that need system-redundancy with high performance, and can tolerate some delaying of data redundancy, Vilamb provides a tunable knob between performance and time-to-coverage. Even with the delayed coverage, Vilamb increases the mean time to data loss due to firmware-induced corruptions by up to two orders of magnitude in comparison to maintaining no system-redundancy.

Tvarak is a software-managed hardware offload to efficiently maintain system-redundancy for direct-access (DAX) NVM storage. Tvarak reconciles the mismatch between DAX granularities and typical system-redundancy granularities by introducing cache-line granular checksums (only) for DAX-mapped data. Tvarak also uses caching to reduce the number of extra NVM accesses for maintaining and verifying system-redundancy. Applications' data access locality leads to reuse of system-redundancy that Tvarak leverages with a small dedicated on-controller cache and configurable LLC partitions. Simulation-based evaluation demonstrates Tvarak's efficiency. For example, Tvarak reduces Redis set-only performance by only 3%.

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.

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

continued on page 18

continued from page 17

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 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.

Jinliang Wei prepares to give his talk on "Training Larger Models on TensorFlow w/o Additional GPUs" at the 2019 PDL Retreat.

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,

continued on page 19

continued from page 18

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.

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
- 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.

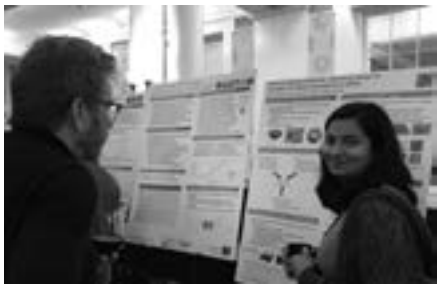
THESIS PROPOSAL: On Building Robustness into Compilation-Based Main-Memory Database Query Engines

Prashanth Menon, SCS
June 17, 2020

Just-in-time (JIT) query compilation is a popular technique to improve analytical query performance in database management systems (DBMSs). However, the cost of compiling each query can be significant relative to its execution time. This overhead prohibits the DBMS from employing well-known adaptive query processing (AQP) methods to re-optimize or generate a new plan for a query if data distributions do not match the optimizer's estimations. The optimizer could eagerly generate multiple sub plans for a query, but it can only include a few alternatives as each addition increases the query compilation time.

In this proposal, we present multiple techniques that bridge the gap between JIT compilation and AQP, with negligible overhead. First, we present Relaxed Operator Fusion (ROF), a first step that enables a compilation-based DBMS to exploit inter-tuple parallelism inherent in a plan using a combination of memory prefetching and SIMD vectorization resulting in improved performance. Next, we present Permutable Compiled Queries (PCQ), a framework that builds upon ROF to allow the DBMS to modify compiled queries without needing to recompile the plan or including all possible variations before the query begins.

We propose to extend our preliminary work in several major directions. First, we aim to design and evaluate an incremental query compilation engine, a method that blurs the line between optimization and code generation. We believe such an approach allows the DBMS to generate more efficient code while also adapting to runtime operating conditions. Next, we investigate lightweight recompilation, a technique that builds upon our incremental query engine to enable more complex adaptive optimizations not supported by PCQ alone. Finally, we propose to explore the design of more sophisticated policies that underlie many of the adaptive optimizations proposed in this work.



Anuva Kulkarni explains her research on "Massive scaling of MASSIF: Algorithm Design for Scalable FFT-based Simulations on GPUs" to Prof. Dave O'Hallaron at the 2019 PDL Retreat.

THESIS PROPOSAL: On Automatic Database Management System Tuning Using Machine Learning

Dana Van Aken, SCS
June 16, 2020

Database management systems (DBMSs) are an important component of any data-intensive application. But tuning a DBMS to perform well is a notoriously difficult task because they have hundreds of configuration knobs that control aspects of their runtime behavior, such as cache sizes and how frequently data is flushed to disk. Getting the right configuration for these knobs is hard because they are not standardized (i.e., sets of knobs for different DBMSs vary), not

continued on page 20

continued from page 19

independent (i.e., changing one knob may alter the effects of others), and not universal (i.e., the optimal configuration depends on the target workload and hardware). Furthermore, as databases grow in both size and complexity, optimizing a DBMS to meet the needs of new applications has surpassed the abilities of even the best human experts. Recent studies using machine learning techniques to automatically configure a DBMS's knobs have shown that such techniques are able to produce high-quality configurations, however, they need a large amount of training data to achieve good results. Collecting this data is costly and time-consuming. In this thesis, we seek to address the challenge of developing effective yet practical techniques for the automatic configuration of DBMSs using machine learning. We show that leveraging knowledge gained from previous tuning efforts to assist in the tuning of others can significantly reduce the amount of time and resources needed to tune a DBMS for a new application.

THESIS PROPOSAL: *Lightweight Preemptible Functions*

Sol Boucher, SCS
April 22, 2020

Modern operating systems provide task preemption as a resource sharing mechanism: when the total number of threads exceeds the number of processors, the kernel scheduler preempts long-running or low-priority threads to allow others to run. Preemption is also useful to applications in its own right, and its interface influences the structure and architecture of such programs. Providing only an asynchronous interface encourages the programmer to leave even simple scheduling to the operating system, thereby accepting the scheduler's overhead and coarse resolution. We introduce a novel abstraction for preemption at the granularity of a synchronous function call, and demonstrate that this represents a more efficient and composable interface that enables new functionality for latency-critical applications, while being both compatible with the existing systems stack and expressive enough to encode classic asynchrony primitives.

THESIS PROPOSAL: Modernizing Models and Management of the Memory Hierarchy for Non-Volatile Memory

Charles McGuffey, SCS
April 21, 2020

Non-volatile memory technologies (NVMs) are a new family of technologies that combine near memory level performance with near storage level cost density. The result is a new type of memory hierarchy layer that exists and performs somewhere between the two. These new technologies offer many opportunities for performance improvements, but bring many of their own unique challenges.

In this thesis, we focus on how the introduction of NVMs affects memory management and caching. Our work is broken into four primary categories. 1. We study how fault tolerance can be achieved at a program level through the use of NVM memory technologies that survive faults. 2. We extend the traditional model of caching to account for the data writes that become more significant in NVM memories. 3. We consider a variant caching model with a cache that works at the granularity of lines above a storage layer that works at block granularity. 4. We approach the increasing difficulty of caching problems in the modern hierarchy using the tools available in information theory to close the gap between theoretical and practical results.

Throughout our work we rely on a blend of theoretical and practical approaches. We provide models for processor faults, cache writebacks, cache-storage communication, and trace complexity that isolate the targeted effects from orthogonal complications. For each model, we show worst case theoretical bounds for our algorithms along with proofs that explain how the benefits are derived. We then take our results and provide empirical evaluations to show their effectiveness in practice. We believe that our ideas and approach provide a solid foundational study on memory hierarchy design in the era of non-volatile memories.

THESIS PROPOSAL: Realizing Value in Shared Compute Infrastructures

Andrew Chung, SCS
February 6, 2020

As operations become increasingly digitized and as data processing tasks become more and more specialized with the proliferation of various types of data applications, companies are moving their workloads off of dedicated, siloed clusters in favor of more cost-efficient shared data infrastructures. These shared data infrastructures are often deployed on highly heterogeneous servers, are multi-tenant with server resources shared across multiple organizations, and serve widely diverse workloads.

Both operators and users of such shared data infrastructures strive to optimize for value.

Operators seek to satisfy the demands of their customers (i.e., help users maximize their value) to increase adoption and lower turnover, all the while without sacrificing cluster operation costs and overhead. At the same time, users look to complete their tasks in an efficient and timely manner without having to pay large amounts of money.

But, the highly heterogeneous nature of these shared environments imposes a high barrier to value attainment for both operators and users: Operators

continued on page 21

continued from page 20

face difficult challenges in knowing how to assign compute resources to customers when heavily loaded. Users, on the other hand, have a wide variety of different types of compute resources available for rent, making it difficult for them to make value-efficient resource acquisition decisions for their applications, given application constraints. Indeed, maximizing value in shared data infrastructures necessarily requires effort from both operators and users.

In my work, I explore the problem of value attainment in shared data infrastructures from both the perspectives of operators and users. On the operator front, I explore using the notions of historic inter-job dependencies and expected job utility to inform cluster resource managers about upcoming jobs, their resource requirements, and the potential value they generate to users. Cluster resource managers can in turn use the information to effectively allocate cluster resources to jobs to achieve high user value attainment. On the user front, my work proposes and evaluates two resource acquisition strategies and systems for renting virtual machine (VM) instances in the public cloud, one for running online services and the other for general batch analytics jobs, with each demonstrating significant cost savings for users.

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.

Abutalib Aghayev explains "Adopting LSM-Trees to Zoned Storage Devices" to the attendees of the 2019 PDL Retreat.

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

continued on page 22

continued from page 21

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.

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 from page 7

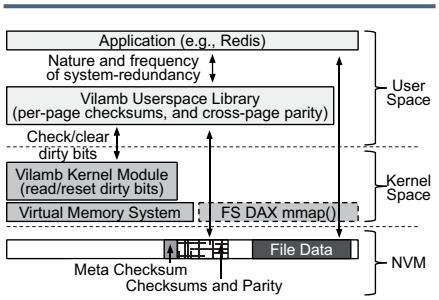
obtained by pulling the arms are functions of a common latent random variable. The correlation between arms due to the common random source can be used to design a generalized upper-confidence-bound (UCB) algorithm that identifies certain arms as non-competitive, and avoids exploring them. As a result, we reduce a K-armed bandit problem to a C + 1-armed problem, where C + 1 includes the best arm and C competitive arms. Our regret analysis shows that the competitive arms need to be pulled O(log T) times, while the non-competitive arms are pulled only O(1) times. As a result, there are regimes where our algorithm achieves a O(1) regret as opposed to the typical logarithmic regret scaling of multi-armed bandit algorithms. We also evaluate lower bounds on the expected regret and prove that our correlated-UCB algorithm achieves O(1) regret whenever possible.

Vilamb: Low Overhead Asynchronous Redundancy for Direct Access NVM

Rajat Kateja, Andy Pavlo, Greg Ganger

Carnegie Mellon University Parallel Data Lab Technical Report CMU-PDL-20-101, April 2020.

Lazy redundancy maintenance can provide direct access non-volatile memory (NVM) with low-overhead data integrity features. The Vilamb library lazily maintains redundancy (per-page checksums and cross-page parity) for applications that exploit fine-grained direct load/store access to NVM data. To do so, Vilamb repurposes page table dirty bits to identify pages where redundancy must be updated, addressing the consistency challenges of using dirty bits across crashes. A periodic background thread updates outdated redundancy at a dataset-specific frequency chosen to tune the performance vs. time-to-coverage tradeoff. This approach avoids critical path interpositioning and often amortizes redundancy updates across many stores to a page, enabling Vilamb to maintain redundancy at just a few percent overhead. For example, MongoDB's YCSB throughput drops by less than 2% when using Vilamb with a 30 sec period and by only 3–7% with a 1 sec period. Compared to the state-of-the-art approach, Vilamb with a 30 sec period increases the throughput by up to 1.8 for Redis with YCSB workloads and by up to 4.2 for write-only micro-benchmarks.



The user space library performs the checksum and parity computations with a period that is set by the application. The kernel module checks and clears the dirty bits when requested by the user space library.

Learning-Based Coded Computation

Jack Kosaian, K.V. Rashmi, Shivaram Venkataraman

IEEE Journal on Selected Areas in Information Theory, March 2020.

Recent advances have shown the potential for coded computation to impart resilience against slowdowns and failures that occur in distributed computing systems. However, existing coded computation approaches are either unable to support non-linear computations, or can only support a limited subset of non-linear computations while requiring high resource overhead. In this work, we propose a learning-based coded computation framework to overcome the challenges of performing coded computation for general non-linear functions. We show that careful use of machine learning within the coded computation framework can extend the reach of coded computation to imparting resilience to more general non-linear computations. We showcase the applicability of learning-based coded computation to neural network inference, a major workload in production services. Our evaluation results show that learning-based coded computation enables accurate reconstruction of unavailable results from widely deployed neural networks for a variety of inference tasks such as image classification, speech recognition, and object localization. We implement our proposed approach atop an open-source prediction serving system and show its promise in alleviating slowdowns that occur in neural network inference. These results indicate the potential for learning-based approaches to open new doors for the use of coded computation for broader, non-linear computations.

The Case for Custom Storage Backends in Distributed Storage Systems

A. Aghayev, S. Weil, M. Kuchnik, M. Nelson, G. Ganger, G. Amvrosiadis

ACM Transactions on Storage, Volume 16, Issue 1, March 2020.

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.

continued on page 24

continued from page 23

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 backward-incompatible storage hardware possible, an important trait in a changing storage landscape that is learning to embrace hardware diversity.

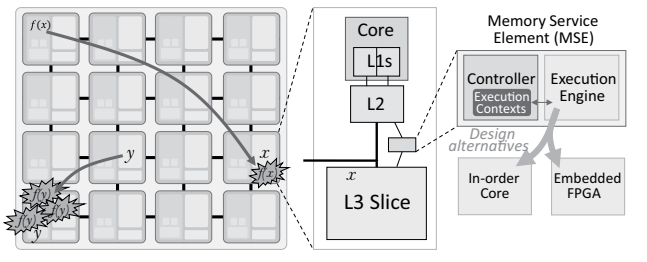
Livia: Data-Centric Computing Throughout the Memory Hierarchy

Elliot Lockerman, Axel Feldmann, Mohammad Bakhshalipour, Alexandru Stanescu, Shashwat Gupta, Daniel Sanchez, Nathan Beckmann

ASPLOS '20: Proceedings of the 25th International Conference on Architectural Support for Programming Languages and Operating Systems, Virtual Lausanne, Switzerland, March 16-20, March 2020.

In order to scale, future systems will need to dramatically reduce data movement. Data movement is expensive in current designs because (i) traditional memory hierarchies force computation to happen unnecessarily far away from data and (ii) processing-in-memory approaches fail to exploit locality.

We propose Memory Services, a flexible programming model that enables data-centric computing throughout the memory hierarchy. In Memory Services, applications express functionality as graphs of simple tasks, each task indicating the data it operates on. We design and evaluate Livia, a new system architecture for Memory Services that dynamically schedules tasks and data at the location in the memory hierarchy that minimizes overall data movement. Livia adds less than 3% area overhead to a tiled multicore and accelerates challenging irregular workloads by 1.3× to 2.4× while reducing dynamic energy by 1.2× to 4.7×.



Livia adds a Memory Service Element (MSE) to each tile in a multicore. MSEs contain a controller that migrates tasks and data to minimize data movement, and an execution engine that efficiently executes tasks near-data. The MSE execution engine is implemented as either a simple in-order core or a small embedded FPGA.

Learning Relaxed Belady for Content Distribution Network Caching

Zhenyu Song, Daniel S. Berger, Kai Li, Wyatt Lloyd

17th USENIX Symposium on Networked Systems Design and Implementation (NSDI '20). February 25–27, 2020. Santa Clara, CA.

This paper presents a new approach for caching in CDNs that uses machine learning to approximate the Belady MIN (oracle) algorithm. To accomplish this complex task, we propose a CDN cache design called Learning Relaxed Belady (LRB) to mimic a Relaxed Belady algorithm, using the concept of Belady boundary. We also propose a metric called good decision ratio to help us make better design decisions. In addition, the paper addresses several challenges to build an end-to-end machine learning caching prototype, including how to gather training data, limit memory overhead, and have lightweight training and prediction.

We have implemented an LRB simulator and a prototype within Apache Traffic Server. Our simulation results with 6 production CDN traces show that LRB reduces WAN traffic compared to a typical production CDN cache design by 4–25%, and consistently outperform other state-of-the-art methods. Our evaluation of the LRB prototype shows its overhead is modest and it can be deployed on today's CDN servers.

Scalable Pointer Analysis of Data Structures using Semantic Models

Pratik Fegade, Christian Wimmer

29th Conference on Compiler Construction (CC '20), February 22–23, 2020, San Diego, CA, USA.

Pointer analysis is widely used as a base for different kinds of static analyses and compiler optimizations. Designing a scalable pointer analysis with acceptable precision for use in production compilers is still an open question. Modern object oriented languages like Java and Scala promote abstractions and code reuse, both of which make it difficult to achieve precision. Collection data structures are an example of a pervasively used component in such languages. But analyzing collection implementations with full context sensitivity leads to prohibitively long analysis times. We use semantic models to reduce the complex internal implementation of, e.g., a collection to a small and concise model. Analyzing the model with context sensitivity leads to precise results with only a modest increase in analysis time. The models must be written manually, which is feasible because a model method usually consists of only a few statements. Our implementation in GraalVM Native Image shows a rise in useful precision (1.35X rise in the number of checkcast statements that

continued on page 25

continued from page 24

can be elided over the default analysis configuration) with a manageable performance cost (19% rise in analysis time).

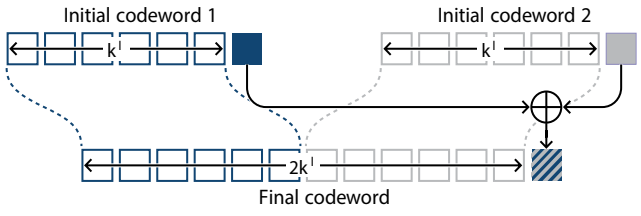
Convertible Codes: New Class of Codes for Efficient Conversion of Coded Data in Distributed Storage

Francisco Maturana, K. V. Rashmi

11th Innovations in Theoretical Computer Science Conference (ITCS 2020). Seattle, WA, January 12-14, 2020.

Erasure codes are typically used in large-scale distributed storage systems to provide durability of data in the face of failures. In this setting, a set of k blocks to be stored is encoded using an [n, k] code to generate n blocks that are then stored on different storage nodes. A recent work by Kadekodi et al. [32] shows that the failure rate of storage devices vary significantly over time, and that changing the rate of the code (via a change in the parameters n and k) in response to such variations provides significant reduction in storage space requirement. However, the resource overhead of realizing such a change in the code rate on already encoded data in traditional codes is prohibitively high.

Motivated by this application, in this work we first present a new framework to formalize the notion of code conversion – the process of converting data encoded with an [nI , kI] code into data encoded with an [nF , kF] code while maintaining desired decodability properties, such as the maximum-distance-separable (MDS) property. We then introduce convertible codes, a new class of code pairs that allow for code conversions in a resource-efficient manner. For an important parameter regime (which we call the merge regime) along with the widely used linearity and MDS decodability constraint, we prove tight bounds on the number of nodes accessed during code conversion. In particular, our achievability result is an explicit construction of MDS convertible codes that are optimal for all parameter values in the merge regime albeit with a high field size. We then present explicit low-field-size constructions of optimal MDS convertible codes for a broad range of parameters in the merge regime. Our results thus show that it is indeed possible to achieve code conversions with significantly lesser resources as compared to the default approach of re-encoding.



Example of code conversion: two codewords of a $[k^I + 1, k^I]$ single-parity-check code become one codeword of a $[2k^I + 1, 2k^I]$ single-parity-check code. The parity symbols are shown shaded. The data symbols from each codeword are preserved, and the parity symbol from the final codeword is the sum of the parities from each initial codeword.

SlowMo: Improving Communication-Efficient Distributed SGD With Slow Momentum

Jianyu Wang, Vinayak Tantia, Nicolas Ballas, Michael Rabbat

ICLR 2020: International Conference on Learning Representations, Apr 26-May 1, 2020, Virtual Addis Ababa, Ethiopia.

Distributed optimization is essential for training large models on large datasets. Multiple approaches have been proposed to reduce the communication overhead in distributed training, such as synchronizing only after performing multiple local SGD steps, and decentralized methods (e.g., using gossip algorithms) to decouple communications among workers. Although these methods run faster than ALLREDUCE-based methods, which use blocking communication before every update, the resulting models may be less accurate after the same number of updates. Inspired by the BMUF method of Chen & Huo (2016), we propose a slow momentum (SLOWMO) framework, where workers periodically synchronize and perform a momentum update, after multiple iterations of a base optimization algorithm. Experiments on image classification and machine translation tasks demonstrate that SLOWMO consistently yields improvements in optimization and generalization performance relative to the base optimizer, even when the additional overhead is amortized over many updates so that the SLOWMO runtime is on par with that of the base optimizer. We provide theoretical convergence guarantees showing that SLOWMO converges to a stationary point of smooth non-convex losses. Since BMUF can be expressed through the SLOWMO framework, our results also correspond to the first theoretical convergence guarantees for BMUF.

Writeback-Aware Caching

Nathan Beckmann, Phillip B. Gibbons, Bernhard Haeupler, Charles McGuffey

Society for Industrial and Applied Mathematics. 2020.

The literature on cache replacement, while both detailed and extensive, neglects to account for the flow of data to storage. Motivated by emerging memory technologies and the increasing importance of memory bandwidth and energy consumption, we seek to fill this gap by studying the Write-back-Aware Caching Problem. This problem modifies traditional caching problems by explicitly accounting for

continued on page 26

continued from page 25

the cost of writing modified data back to memory on eviction.

In the offline setting with maximum writeback cost $\omega > 0$, we show that writeback-aware caching is NP-complete and Max-SNP hard. Moreover, we show that Furthest-in-the-Future, the optimal deterministic policy when ignoring writebacks, is only $(\omega + 1)$-competitive. These negative results hold even for the simple variant of the problem in which data items have unit size, unit miss cost, and unit writeback cost ($\omega = 1$). To overcome this difficulty, we provide practical algorithms to compute upper and lower bounds for the optimal policy on real traces. In the online setting, we present a deterministic replacement policy called Writeback-Aware Landlord and show that it obtains the optimal competitive ratio. Our bounds on the optimal offline policy and our optimal competitive ratio hold even for the most general variant in which data items have variable sizes, variable miss costs, and variable writeback costs. Finally, we perform an experimental study on real-world traces showing that Writeback-Aware Landlord outperforms state-of- the-art cache replacement policies when writebacks are costly, thereby illustrating the practical gains of explicitly accounting for writebacks.

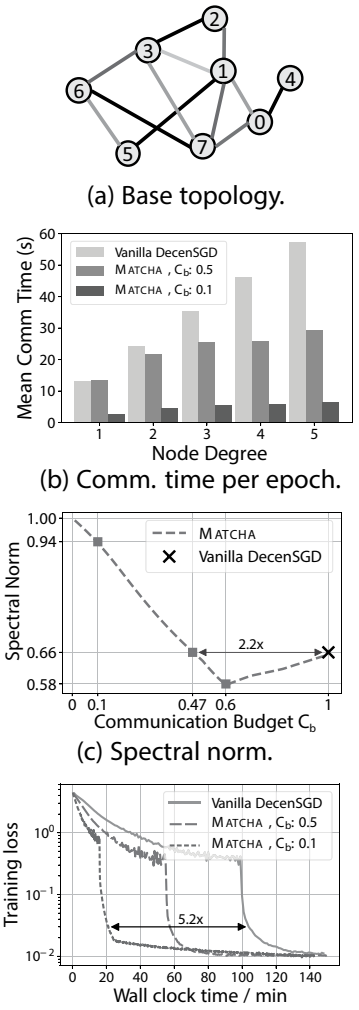
MATCHA: Speeding Up Decentralized SGD via Matching Decomposition Sampling

Jianyu Wang, Anit Sahu, Gauri Joshi, Soummya Kar

NeurIPS Workshop of Federated Learning for Data Privacy and Confidentiality, Dec 13, 2019. Vancouver, BC, Canada.

Distinguished Student Paper Award

This paper studies the problem of error-runtime trade-off, typically encountered in decentralized training based on stochastic gradient descent (SGD) using a given network. While a denser (sparser) network topology results in faster (slower) error convergence in terms of iterations, it incurs more (less) communication time/delay per iteration. In this paper, we propose MATCHA, an algorithm that can achieve a win-win in this error-runtime trade-off for any arbitrary network topology. The main idea of MATCHA is to parallelize inter-node communication by decomposing the topology into matchings. To preserve fast error convergence speed, it identifies and communicates more frequently over critical links, and saves communication time by using other links less frequently. Experiments on a suite of datasets and deep neural networks validate the theoretical analyses and demonstrate that MATCHA takes up to 5× less time than vanilla decentralized SGD to reach the same training loss.



Comparison of vanilla decentralized SGD (DecenSGD) and MATCHA. (a) An example base topology generated using Erdos-Rényi model. (b) MATCHA reduces the communication time non-uniformly. Nodes with higher degrees (for instance node 1 with degree 5) tend to have more redundant links. (c) Lower spectral norm yields better convergence rate. Communication budget C_b represents the average frequency of communication over the links in the network. (d) Loss-versus-time curves when training WideResNet on CIFAR-100.

Rateless Codes for Near-Perfect Load Balancing in Distributed Matrix-vector Multiplication

Ankur Mallick, Malhar Chaudhari, Ganesh Palanikumar, Utsav Sheth, Gauri Joshi

Proc. ACM Meas. Anal. Comput. Syst., Vol. 3, No. 3, Article 58. Publication date: December 2019. Best Paper at ACM Sigmetrics, June 8-12, 2020, Boston, MA.

Large-scale machine learning and data mining applications require computer systems to perform massive matrix-vector and matrix-matrix multiplication operations that need to be parallelized across multiple nodes. The presence of straggling nodes – computing nodes that unpredictably slowdown or fail – is a major bottleneck in such distributed computations. Ideal load balancing strategies that dynamically allocate more tasks to faster nodes require knowledge or monitoring of node speeds as well as the ability to quickly move data. Recently proposed fixed-rate erasure coding strategies can handle unpredictable node slowdown, but they ignore partial work done by straggling nodes thus resulting in a lot of redundant computation. We propose a rateless fountain coding

continued on page 27

continued from page 26

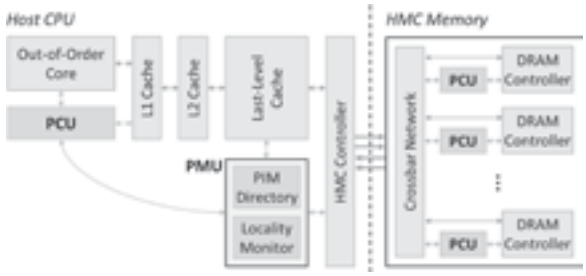
strategy that achieves the best of both worlds – we prove that its latency is asymptotically equal to ideal load balancing, and it performs asymptotically zero redundant computations. Our idea is to create linear combinations of the m rows of the matrix and assign these encoded rows to different worker nodes. The original matrix-vector product can be decoded as soon as slightly more than m row-vector products are collectively finished by the nodes. We conduct experiments in three computing environments: local parallel computing, Amazon EC2, and Amazon Lambda, which show that rateless coding gives as much as 3× speed-up over uncoded schemes.

Processing-in-Memory: A Workload-Driven Perspective

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

IBM Journal of Research and Development (JRD), Vol. 63, No. 6, November/December 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, many practical system-level issues 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 toward 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 in programming these applications for PIM architectures. Third, we describe challenges that remain for the widespread adoption of PIM.



Example architecture for PIM-enabled instructions.

MANIC: A Vector-Dataflow Architecture for Ultra-Low-Power Embedded Systems

G Gobieski, A Nagi, N Serafin, MM Isgenc, N Beckmann, B Lucia

MICRO '52: Proceedings of the 52nd Annual IEEE/ACM International Symposium on Microarchitecture, Columbus, OH, October 2019.

Ultra-low-power sensor nodes enable many new applications and are becoming increasingly pervasive and important. Energy efficiency is the key determinant of the value of these devices: battery-powered nodes want their battery to last, and nodes that harvest energy should minimize their time spent recharging. Unfortunately, current devices are energy-inefficient. In this work, we present MANIC, a new, highly energy-efficient architecture targeting the ultra-low-power sensor domain. MANIC achieves high energy-efficiency while maintaining programmability and generality. MANIC introduces vector-dataflow execution, allowing it to exploit the dataflows in a sequence of vector instructions and amortize instruction fetch and decode over a whole vector of operations. By forwarding values from producers to consumers, MANIC avoids costly vector register file accesses. By carefully scheduling code and avoiding dead register writes, MANIC avoids costly vector register writes. Across seven benchmarks, MANIC is on average 2.8× more energy efficient than a scalar baseline, 38.1% more energy-efficient than a vector baseline, and gets to within 26.4% of an idealized design.

Demystifying Complex Workload–DRAM Interactions: An Experimental Study

S. Ghose, T. Li, N. Hajinazar, D. Senol Cali, O. 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), Vol. 3, No. 3, December 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

continued on page 28

continued from page 27

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 multiprogrammed 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.

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.

Sol Boucher (upper left) celebrates with friends Chris Canel and Angela Jiang after successfully completing his thesis proposal over Zoom.

Vantage: Optimizing Video Upload for Time-shifted Viewing of Social Livestreams

Devdeep Ray, Jack Kosaian, K. V. Rashmi, Srini Seshan.

ACM SIGCOMM, August 19-24, 2019, Beijing, China.

Social live video streaming (SLVS) applications are becoming increasingly popular with the rise of platforms such as Facebook- Live, YouTube-Live, Twitch and Periscope. A key characteristic that differentiates this new class of applications from traditional live streaming is that these live streams are watched by viewers at different delays; while some viewers watch a live stream in real-time, others view the content in a time-shifted manner at different delays. In the presence of variability in the upload bandwidth, which is typical in mobile environments, existing solutions silo viewers into either receiving low latency video at a lower quality or a higher quality video with a significant delay penalty, without accounting for the presence of diverse time-shifted viewers.

In this paper, we present Vantage, a live-streaming upload solution that improves the overall quality of experience for diverse time-shifted viewers by using selective quality-enhancing retransmissions in addition to real-time frames, optimizing the encoding schedules to balance the allocation of the available bandwidth between the two. Our evaluation using real-world mobile network traces shows that Vantage can provide high quality simultaneously for both low-latency and delayed viewing. For delayed viewing, Vantage achieves an average improvement of 19.9% over real-time optimized video streaming techniques across all the network traces and test videos, with observed gains of up to 42.9%. These benefits come at the cost of an average drop in realtime quality of 3.3%, with a maximum drop

continued on page 29

continued from page 28

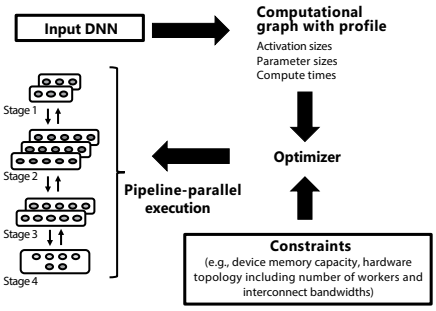
of 7.1%. This represents a significant performance improvement over current techniques used for SLVS applications, which primarily optimize the video upload for real-time viewing.

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.



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.

Automating Dependence-Aware Parallelization of Machine Learning Training on Distributed Shared Memory

Jinliang Wei, Garth A. Gibson, Phillip B. Gibbons, Eric P. Xing

EuroSys '19: Proceedings of the Fourteenth EuroSys Conference, March 2019, Dresden, Germany.

Machine learning (ML) training is commonly parallelized using data parallelism. A fundamental limitation of data parallelism is that conflicting (concurrent) parameter accesses during ML training usually diminishes or even negates the benefits provided by additional parallel compute resources. Although it is possible to avoid conflicting parameter accesses by carefully scheduling the computation, existing systems rely on programmer manual parallelization and it remains a question when such parallelization is possible. We present Orion, a system that automatically parallelizes serial imperative ML programs on distributed shared memory. The core of Orion is a static dependence analysis mechanism that determines when dependence-preserving parallelization is effective and maps a loop computation to an optimized distributed computation schedule. Our evaluation shows that for a number of ML applications, Orion can parallelize a serial program while preserving critical dependences and thus achieve a significantly faster convergence rate than data-parallel programs and a matching convergence rate and comparable computation throughput to state-of-the-art manual parallelizations including model-parallel programs.

External vs. Internal: An Essay on Machine Learning Agents for Autonomous Database Management Systems

Andrew Pavlo, Matthew Butrovich, Ananya Joshi, Lin Ma, Prashanth Menon, Dana Van Aken, Lisa Lee, Ruslan Salakhutdinov

Bulletin of the IEEE Computer Society Technical Committee on Data Engineering, 42(2): 32-46 (2019)

The limitless number of possible ways to configure database management systems (DBMSs) has rightfully earned them the reputation of being difficult to manage and tune. Optimizing a DBMS to meet the needs of an application has surpassed the abilities of humans. This is because the correct configuration of a DBMS is highly dependent on a number of factors that are beyond what humans can reason about. The problem is further exacerbated in large-scale deployments with thousands or even millions of individual DBMS installations that each have their own tuning requirements.

To overcome this problem, recent research has explored using machine learning-based (ML) agents for automated tuning of DBMSs. These agents extract performance metrics and be-

continued on page 30

continued from page 29

havioral information from the DBMS and then train models with this data to select tuning actions that they predict will have the most benefit. They then observe how these actions affect the DBMS and update their models to further improve their efficacy. In this paper, we discuss two engineering approaches for integrating ML agents in a DBMS. The first is to build an external tuning controller that treats the DBMS as a black-box. The second is to integrate the ML agents natively in the DBMS's architecture. We consider the trade-offs of these approaches in the context of two projects from Carnegie Mellon University (CMU).

Non-Volatile Memory Database Management Systems

Joy Arulraj, Andrew Pavlo

Synthesis Lectures on Data Management, Morgan & Claypool Publishers, 2019.

This book explores the implications of non-volatile memory (NVM) for database management systems (DBMSs). The advent of NVM will fundamentally change the dichotomy between volatile memory and durable storage in DBMSs. These new NVM devices are almost as fast as volatile memory, but all writes to them are persistent even after power loss. Existing DBMSs are unable to take full advantage of this technology because their internal architectures are predicated on the assumption that memory is volatile. With NVM, many of the components of legacy DBMSs are unnecessary and will degrade the performance of data-intensive applications.

We present the design and implementation of DBMS architectures that are explicitly tailored for NVM. The book focuses on three aspects of a DBMS: (1) logging and recovery, (2) storage and buffer management, and (3) indexing. First, we present a logging and recovery protocol that enables the DBMS to support near-instantaneous recovery.

Greg Ganger hands Angela Jiang her virtual diploma in the presence of her thesis committee following the completed defense over Zoom of her dissertation on "Improving Deep Learning Training and Inference with Dynamic Hyperparameter Optimization."

Second, we propose a storage engine architecture and buffer management policy that leverages the durability and byte-addressability properties of NVM to reduce data duplication and data migration. Third, the book presents the design of a range index tailored for NVM that is latch-free yet simple to implement. All together, the work described in this book illustrates that rethinking the fundamental algorithms and data structures employed in a DBMS for NVM improves performance and availability, reduces operational cost, and simplifies software development.

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 31

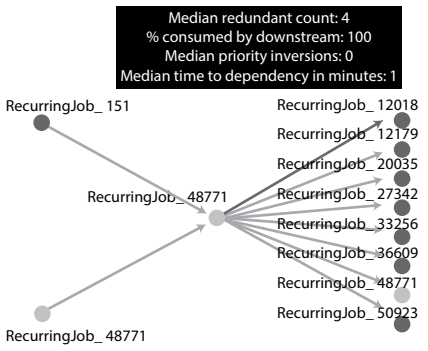
continued from page 30

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.

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, 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.

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.

continued on page 32

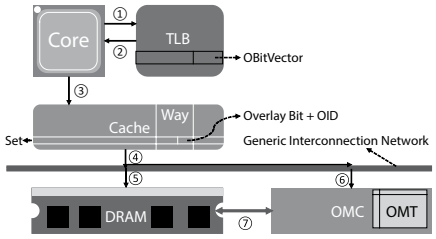
continued from page 31

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 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.



Page Overlays – This diagram shows how memory system handles overlay. 1) TLB lookup using VA; 2) TLB outputs either translated PA or VA based on OBitVector; 3) Cache lookup using TLB output; 4) On cache miss (or eviction), include (OID, VA) in the fetch (eviction) request; 5) If address is PA (or not an overlay cache line), send request to the main memory; 6) If address is VA (or is an overlay cache line), send request to OMC; 7) OMC queries OMT using (OID, VA), obtains PA, and fetches from memory.

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.

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, becomes 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.

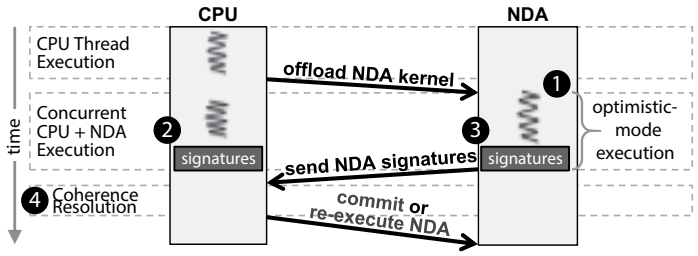
Based on our observations, we propose CoNDA, a coherence mechanism that lets an NDA optimistically execute an NDA kernel, under the assumption that the NDA has all necessary coherence permissions. This optimistic execution allows CoNDA to gather information on the memory accesses performed by the NDA 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

continued on page 33

RECENT PUBLICATIONS

continued from page 32



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).

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.

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.

YEAR IN REVIEW

continued from page 4

January 2020

- Francisco Maturana presented "Convertible Codes: New Class of Codes for Efficient Conversion of Coded Data in Distributed Storage" at the 11th Innovations in Theoretical Computer Science Conference (ITCS 2020) in Seattle, WA.
- Rashmi Vinayak received the Prof. R. Narasimhan Memorial Lecture Award.
- Juncheng (Jason) Yang received a Facebook Fellowship.
- Saurabh Kadekodi gave his speaking skills talk "Gotta have HeART: Improving Storage Efficiency by Exploiting Disk-reliability Heterogeneity."
- Conglong Li proposed his PhD research on "Learned Adaptive Accuracy-Cost Optimization for Machine Learning Systems."

December 2019

- Jianyu Wang presented "MATCHA: Speeding Up Decentralized SGD via Matching Decomposition Sampling" at the NeurIPS workshop of Federated Learning for

continued on page 34

continued from page 33

Data Privacy and Confidentiality, in Vancouver, BC, Canada.

- Aurick Qiao gave his speaking skills talk "Challenges and Opportunities in Dynamic Resource Scheduling for Distributed Deep Learning."

November 2019

- Jack Kosaian gave his speaking skills talk on "Parity Models: Erasure-Coded Resilience for Prediction Serving Systems."
- Jason Yang gave his speaking skills talk on "C2DN: How to Code on the Edge for Content Delivery Network."

October 2019

- The Delta FS Project brought home an R&D 100 Award!
- Aaron Harlap co-authored "PipeDream: Generalized Pipeline Parallelism for DNN Training" for SOSP '19, in Huntsville, ON.
- Abutalib Aghayev presented "File Systems Unfit as Distributed Storage Backends: Lessons from 10 Years of Ceph Evolution" at SOSP '19, in Huntsville, ON, Canada.
- Jack Kosaian presented "Parity Models: Erasure-Coded Resilience for Prediction Serving Systems" at SOSP '19, in Huntsville, ON, Canada.
- Graham Gobieski presented "MANIC: A Vector-Dataflow Architecture for Ultra-Low-Power Embedded Systems" at MICRO '52 in Columbus, OH.
- Nandita Vijaykumar defended her dissertation on "Enhancing Programmability, Portability, and Performance with Rich Cross-Layer Abstractions."
- Huanchen Zhang defended his PhD research on "Memory-Efficient Search Trees for Database Management Systems."

September 2019

- Nathan Beckmann earned an NSF Early CAREER Award.
- Qing Zheng presented "Compact Filters for Fast Online Data Partitioning" at IEEE CLUSTER 2019, in Albuquerque, New Mexico, USA.
- Abutalib Aghayev was awarded the Hima and Jive Graduate Fellowship.
- Lorrie Cranor was made a member of ACM's New Technology Policy Council.
- Amirali Boroumand proposed his PhD research on "Practical Mechanisms for Reducing Processor-Memory Data Movement in Modern Workloads."
- Ziqi Wang presented "Multi-versioned Page Overlays: Enabling Faster Serializable Hardware Transactional Memory" at the 28th Parallel Architecture and Compiler Technologies (PACT'19), in Seattle, WA.
- Damla Senol proposed her PhD thesis on "Accelerating Genome Sequence Analysis via Efficient Hardware-Algorithm Co-Design."
- Kevin Hsieh defended his PhD thesis on "Machine Learning Systems for Highly-Distributed and Rapidly-Growing Data."

August 2019

- Anuj Kalia defended his dissertation on "Efficient Remote Procedure Calls for Datacenters."
- Rashmi Vinayak's research team presented "Vantage: Optimizing Video Upload for Time-shifted Viewing of Social Livestreams" at ACM SIGCOMM, in Beijing.
- Joy Arulraj, Andrew Pavlo published a book on "Non-Volatile Memory Database Management Systems: Synthesis Lectures on Data Management," with Morgan & Claypool Publishers.

July 2019

- Dong Zhou defended his dissertation on "Data Structure Engineering for High Performance Software Packet Processing."
- Jin Kyu Kim presented "STRADS-AP: Simplifying Distributed Machine Learning Programming without Introducing a New Programming Model" at the 2019 USENIX Annual Technical Conference, in Renton, WA.
- Ankur Mallick and Gauri Joshi presented "Rateless Codes for Distributed Computations with Sparse Compressed Matrices" at the IEEE International Symposium on Information Theory (ISIT), in Paris, France.
- Andrew Chung presented "Peering through the Dark: An Owl's View of Inter-job Dependencies and Jobs' Impact in Shared Clusters" at SIGMOD '19, in Amsterdam, Netherlands.

June 2019

- Abutalib Aghayev proposed his PhD research on "Efficiently Adopting Zone Devices in Distributed Storage."
- Amarali Boroumand presented "CoNDA: Efficient Cache Coherence Support for Near-Data Accelerators" at ISCA, in Phoenix, AZ.
- Saugata Ghose presented "Understanding Interactions of Workloads and DRAM Types: A Comprehensive Experimental Study" and "Demystifying Complex Workload–DRAM Interactions: An Experimental Study" at the ACM SIGMETRICS/IFIP Performance Conference, in Phoenix, AZ.
- Qing Zheng proposed his thesis research on "Distributed Metadata and Streaming Data Indexing as Scalable Filesystem Services."

May 2019

- 21st Annual PDL Spring Visit Day.