Amogh **Akshintala**

137A Purefoy Road, Chapel Hill, NC 27514

+1 (631)-371-4820 | aakshintala@cs.unc.edu | aakshintala.com | aakshintala

Education

Doctor of Philosophy | Computer Science

University of North Carolina - Chapel Hill — *Aug 2016 — Dec 2020 (expected)*

Stony Brook University — *Jan 2014 — Aug 2016*

Master of Science | Computer Science

Stony Brook University | GPA: 3.610 — *Aug 2012 — Dec 2013 (issued May 2016)*

Bachelor of Engineering | Computer Science and Engineering

Visvesvaraya Technological University | First Class with Distinction — *2008 — 2012*

Publications

Conferences

Accelerated Virtualization of Accelerators

Hangchen Yu, Arthur Peters, **Amogh Akshintala**, Christopher J. Rossbach — *ASPLOS '20*

Trillium: The code is the IR

Amogh Akshintala, Hangchen Yu, Arthur Peters, Christopher J. Rossbach — *VIRT '19*

x86-64 Instruction Usage among C/C++ Applications

Amogh Akshintala, Bhushan P. Jain, Chia-che Tsai, Michael Ferdman, Donald E. Porter — *SYSTOR '19*

Optimizing Every Operation in a Write-optimized File System — *Awarded Best Paper*

Jun Yuan, Yang Zhan, William Jannen, Prashant Pandey, **Amogh Akshintala**, Kanchan Chandnani, Pooja Deo, Zardosht Kasheff, Leif Walsh, Michael Bender, Martin Farach-Colton, Rob Johnson, Bradley C. Kuszmaul, and Donald E. Porter. — *FAST '16*

BetrFS: A Right-Optimized Write-Optimized File System — *Best Paper Runner-up*

William Jannen, Jun Yuan, Yang Zhan, **Amogh Akshintala**, John Esmet, Yizheng Jiao, Ankur Mittal, Prashant Pandey, Phaneendra Reddy, Leif Walsh, Michael Bender, Martin Farach-Colton, Rob Johnson, Bradley C. Kuszmaul, and Donald E. Porter. — *FAST '15*

Journals

Writes Wrought Right, and Other Adventures in File System Optimization.

Jun Yuan, Yang Zhan, William Jannen, Prashant Pandey, **Amogh Akshintala**, Kanchan Chandnani, Pooja Deo, Zardosht Kasheff, Leif Walsh, Michael A. Bender, Martin Farach-Colton, Rob Johnson, Bradley C. Kuszmaul, and Donald E. Porter. — *ACM Transactions on Storage (TOS) Mar'17*

BetrFS: Write-Optimization in a Kernel File System

William Jannen, Jun Yuan, Yang Zhan, **Amogh Akshintala**, John Esmet, Yizheng Jiao, Ankur Mittal, Prashant Pandey, Phaneendra Reddy, Leif Walsh, Michael Bender, Martin Farach-Colton, Rob Johnson, Bradley C. Kuszmaul, and Donald E. Porter. — *ACM Transactions on Storage (TOS) Nov'15*

Workshops

USETL: Unikernels for Serverless Extract Transform and Load. Why should you settle for less? — *Awarded Best Paper*

Henrique Fingler, **Amogh Akshintala**, Christopher J. Rossbach — *APSys '19*

Automatic Virtualization of Accelerators

Hangchen Yu, Arthur Peters, **Amogh Akshintala**, and Christopher J. Rossbach. — *HotOS '19*

Talk to My Neighbors Transport: Decentralized Data Transfer and Scheduling Among Accelerators.

Amogh Akshintala, Vance Miller , Donald E. Porter, and Christopher J. Rossbach. — *SFMA '18*

Experience

Internships

VMware Research Group

Research Intern — *Summer 2018*

Research Intern — *Summer 2017*

VMware Inc.

Member of Technical Staff - Intern — *Summer 2014*

Tintri Inc.

Member of Technical Staff - Intern — *Summer 2013*

Academic

Lecturer (limited-term)

Computer Architecture

University of North Carolina at Chapel Hill — *Jan 2020 — May 2020*

Research Assistant

University of North Carolina at Chapel Hill — *Aug 2016 — present*

Stony Brook University — *Jan 2015 — Aug. 2016*

Visiting Researcher

University of Texas at Austin — *Fall 2017; Calendar Year 2019*

Teaching Assistant

Stony Brook University — Operating Systems (Graduate) | Network Security (Graduate) — *Spring & Fall 2014*

Service

External Reviewer

ACM SOCC '18

Projects

Accelerator Virtualization

Sprint '18 - ongoing

- Accelerators seem to operate as horizontally and vertically isolated silos — i.e., the only exposed surfaces that can be interposed on are either the user-space API or the HW interface. Leveraging this key insight, we're building an accelerator-virtualization framework that automatically generates much of the code required to forward user-space accelerator APIs through the hypervisor in order to provide the desirable properties of virtualization.
- Outcome: 1 workshop paper. 2 conference papers.

TMNT: accelerating data movement among accelerators

Spring '18 - Spring '19

- Data movement is a first-order determinant of performance when programming high-throughput accelerators. When you throw in the additional challenge of co-ordination among multiple accelerators, the problem is compounded because of the synchronous nature of control in the Master-Slave model that most accelerators operate under.
- We're designing an capability-based hardware structure that provides the necessary primitives to express data placement/movement, manage co-ordination and scheduling of computation on accelerators, and enforce capabilities for processes running on accelerators. We believe these extensions naturally fit the data-flow programming paradigm, which should greatly ease programmability.
- Outcome: 1 Workshop paper.

GPGPU Virtualization

Summer '17 - Spring '18

- Many methods have been proposed to virtualize general purpose compute on GPUs. However, none of them hit the right spot. We observe that there are actually two separate elements that must be virtualized when dealing with GPGPUs: device control, and compute. We extend the para-virtual model used by VMware to investigate our hypothesis that handling these two elements separately is the key to achieving good GPGPU performance.
- I built an LLVM backend for TGSI (VMware and Linux vISA for graphics) that is available on my GitHub (llvm, clang, libclc, mesa).
- Outcome: 1 conference paper

Instruction Popularity

Summer '16 — Fall '19

- Overlapping-ISA multi-core computers have been actively studied in the past decade. However, in most of these studies the data, used to craft the various subsets of the ISA, is collected from a small number of applications, usually popular, and often outdated, benchmarks.
- We performed static analysis on 9000 C/C++ binaries in the Ubuntu 16.04 repositories to get a higher fidelity understanding of static instruction distribution among applications.
- Outcome: 1 Conference paper. Visualization tool: x86instructionpop.com

BetrFS (http://www.betrfs.org/)

Spring '14 — Spring '16

- The B$^\epsilon$–tree File System, or BetrFS, is an in-kernel file system that uses B$^\epsilon$–trees to organize on-disk storage. B$^\epsilon$–trees are a form of write-optimized dictionaries, and offer the same asymptotic behavior for sequential I/O and point queries as a B-tree. The advantage of a B$^\epsilon$–tree is that it can also ingest small, random writes 1-2 orders of magnitude faster than B-trees and other standard on-disk data structures. The goal of BetrFS is to realize performance that strictly dominates the performance of current, general-purpose file systems.
- I was the Benchmark Czar for the project — I was responsible for measuring and understanding performance of BetrFS.
- Outcome: 2 Conference and 2 Journal papers. Project still ongoing; my involvement is limited.

Investigating Next Page Prefetcher behaviour for virtual machines

Spring 2014 - Summer 2015

- Significant differences in the impact of prefetching on the performance of applications running in a VM (vs native) triggered an investigation of the behaviour of the Next Page Prefetcher, which had just been incorporated into Intel CPUs.
- Outcome: We were unable to discern the source of the variation, but I learnt a lot about caches, prefetching, and profiling tools.

Reduced x86_64 backend for LLVM compiler

Spring 2013 - Spring 2015

- As a first step towards the exploring the design-space of ISA-homogeneous, Capability-heterogenous multi-core processors, we attempted to build a backend for the LLVM Compiler that operated on a configurable subset of x86_64 ISA.
- I mentored 2 M.S. students working on this project.