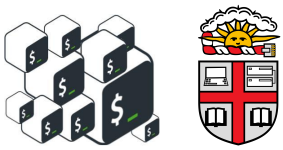

3rd PaSh Research Workshop

Oct. 18–20, 2023 | Brown University, Providence, RI

Data processing needs are skyrocketing. Shell scripting has long been a staple in the data processing toolkit, thanks to its simplicity, universality, and powerful file manipulation capabilities. PaSh is an open-source software project—led by Brown, worked on by several institutions, and hosted by the Linux Foundation—for accelerating, parallelizing, and distributing shell programs to process massive amounts of data. Notably, optimizing programs with PaSh requires minimal-to-zero developer effort and guarantees correctness and compatibility.

This 3-day multi-institutional workshop aims to catalyze research and development on and around PaSh: compiler internals, the scalable distribution extensions, the out-of-order speculative execution engine, formalization, use cases, and components targeting new areas — such as serverless computing and workflow engineering. Day 1 will focus on talks from (and discussions with) core PaSh members; days 2 and 3 are breakout sessions — brainstorming, problem solving, hacking, etc. Sessions marked as "Buffer / Hacking" are to be decided during the workshop, based on areas of focus and priority as determined by the group.

Day 1: Talks

Wed. Oct. 18	**DSI Room 302 (use stairs) & Zoom**
9:00	Agenda & Overview (+coffee/snacks)
9:10	Formal Support for the POSIX Shell
9:30	**Deep Dive: The PaSh Just-in-Time (JIT) Optimization Engine**
10:30	Modeling PaSh's Core Transformations
11:00	dCOZ: Distributed Causal Profiling
11:30	*Break: Lunch*
12:00	**Deep Dive: DiSh & Dynamic Shell-Script Distribution**
1:00	Fault Tolerance for Distributed Shell Script Execution
1:20	Inside the Computational Machine Room
1:40	Bash-Compliant PaSh
2:00	Towards Serverless Shell Scripting
2:30	From Shell Programs to Data Engineering Airflows, Automatically
3:00	*Break: snacks*

3:30 **Deep Dive: Zero-Touch Out-of-Order Shell Script Execution**

4:30 Try: Inspecting command side-effects before modifying systems

5:00 Verifying an Out-of-Order Shell-Script Scheduler

5:15 Towards a Type System for Shell Programs

5:30 Optimizing PaSh: Adaptive Parameter Selection in Diverse Computing Environments

6:00 Discussion: Benchmarks, Workloads, and Evaluation

Day 2: Breakouts

Thu. Oct. 19		Breakout 1	Breakout 2
		CIT 477 and Zoom	**CIT 555**
9:00	9:15	Breakfast & Planning	
9:15	10:30	Benchmarks, Workloads, Evaluation	
10:30	11:30	Order-aware Dataflow Formalization	Benchmarks/Use Cases
11:30	12:30	Distributed Fault Tolerance	Documentation
12:30		*Lunch @ Lubrano*	
		SciLi 804 & Zoom	**CIT 555**
2:00	3:30	Type System	Hacking
3:30	5:00	Buffer / Hacking	TSC Meeting
5:00	6:30	Buffer / Hacking	Buffer / Hacking

Day 3: Breakouts

Fri. Oct. 20		Breakout 1	Breakout 2
		CIT 410 & Zoom	**CIT 555**
9:00	9:15	Breakfast & Planning	
9:15	10:30	Parameter Optimization	Shell Tooling
10:30	11:45	Static Performance Analysis	Bash Parsing
12:00	1:00	*Lunch @ **CIT 368** and on Zoom — talk by Lukas Lazarek*	
		ERC 350 & Zoom	**CIT 555**
1:30	3:30	Scheduling Formalization / Benchs	Bash Parsing
3:30	5:00	Buffer / Hacking	Buffer / Hacking

Agenda & Overview

Nikos Vasilakis
Brown University

Data processing needs are skyrocketing. PaSh is an open-source software project—led by Brown, worked on by several institutions, and hosted by the Linux Foundation—for accelerating, parallelizing, and distributing shell programs that process massive amounts of data. Notably, when optimizing programs, PaSh requires minimal-to-zero developer effort and offers key correctness and compatibility guarantees.

This 3-day multi-institutional workshop focuses on research and development on and around PaSh: compiler internals, the scalable distribution extensions, the out-of-order speculative execution engine, formalization, use cases, and components targeting new areas — such as serverless computing and workflow engineering. The full schedule is at https://binpa.sh

Speaker bio: Nikos Vasilakis is on the faculty of the Computer Science department at Brown University. His research encompasses systems, programming languages, and security—focusing on automatically enhancing software systems with new capabilities. This enhancement forms a transition path towards secure distributed environments targeting multiple scales. Before joining Brown University, Nikos was a research scientist at MIT. He received his M.Sc. and Ph.D. degrees in Computer and Information Science from the University of Pennsylvania.

Formal Support for the POSIX Shell

Michael Greenberg
Stevens Institute of Technology

The POSIX shell is a widely deployed, powerful tool for managing computer systems. The shell is the expert's control panel, a necessary tool for configuring, compiling, installing, maintaining, and deploying systems. Even though it is a powerful, critical infrastructure, the POSIX shell is maligned and misunderstood. Its power and its subtlety make for a dangerous combination. How can we support the POSIX shell? I'll describe a recent line of work on Smoosh, a formal, mechanized, executable small-step semantics for the POSIX shell, with a focus on what constitutes a 'contribution'.

Speaker Bio: Michael Greenberg is an assistant professor at Stevens Institute of Technology. He received BAs in Computer Science and Egyptology from Brown University (2007) and his PhD in Computer Science from the University of Pennsylvania (2013); previously, he was an assistant professor at Pomona College. His work has ranged from functional-reactive JavaScript (with Shriram Krishnamurthi at Brown) to runtime verification of higher-order programs (with Benjamin Pierce at Penn) to software-defined networking (with Dave Walker at Princeton) to present activities focused on the POSIX shell and executable formalism.

Deep Dive: The PaSh Just-in-Time (JIT) Optimization Engine

Konstantinos Kallas
The University of Pennsylvania

Recent shell-script parallelization systems enjoy mostly automated parallel speedups by compiling scripts ahead-of-time. Unfortunately, such static parallelization is hampered by the dynamic behaviors pervasive in shell scripts—e.g., variable expansion and command substitution—which often requires reasoning about the current state of the shell and filesystem. We present a just-in-time (JIT) shell-script compiler, PaSh-JIT, that intermixes evaluation and parallelization during a script's run-time execution. JIT parallelization collects run-time information about the system's state, but must not alter the behavior of the original script and must maintain minimal overhead. PaSh-JIT addresses these challenges by (1) using a dynamic interposition framework, guided by a static preprocessing pass, (2) developing runtime support for transparently pausing and resuming shell execution; and (3) operating as a stateful server, communicating with the current shell by passing messages—all without requiring modifications to the system's underlying shell interpreter. When run on a wide variety of benchmarks, including the POSIX shell test suite, PaSh-JIT (1) does not break scripts, even in cases that are likely to break shells in widespread use; and (2) offers significant speedups, whenever parallelization is possible. These results show that PaSh-JIT can be used as a drop-in replacement for any non-interactive shell use, providing significant speedups without any risk of breakage.

Speaker Bio: Konstantinos Kallas is a final year PhD student at the University of Pennsylvania working with Rajeev Alur. His area of interest is the intersection of computer systems and programming languages. Together with several amazing people Konstantinos has been working on invigorating the research on the shell. The first paper in this line of work, introducing PaSh, a system for parallelizing shell scripts, received the best paper award at EuroSys 2021. He has also published papers improving the shell at ICFP 2021, HotOS 2021, OSDI 2022, NSDI 2023, HotOS 2023. Konstantinos is also interested in programming models and runtimes for stateful serverless computing. He has worked on Azure Durable Functions and has published papers on this topic in OOPSLA 2021, VLDB 2022, and POPL 2023.

Deep Dive: DiSh & Dynamic Shell-Script Distribution

Tammam Mustafa
Google

This talk is a deep walkthrough of DiSh, a PaSh extension for distributing the execution of dynamic shell scripts operating on distributed filesystems such as HDFS. DiSh is designed as a shim that applies program analyses and transformations to leverage distributed computing, while delegating all execution to the underlying shell available on each computing node. As a result, DISH does not require modifications to shell scripts and maintains compatibility with respect to both existing shells and legacy functionality. The talk will dive into both the compilation extensions developed to support DiSh and several runtime components necessary to support the scalable execution of distributed shell programs.

Speaker bio: Tammam Mustafa is a software engineer at Google with a B.S. and MEng in Computer Science from the Massachusetts Institute of Technology (MIT). While pursuing his undergraduate degree, Tammam developed a deep interest in performance engineering and distributed systems. During his M.Eng. program, he worked on improving the performance of PaSh and led the development of DiSh, the PaSh extensions for distributed execution. Currently, in his free time, he is exploring AI/ML workloads, with a particular interest in performance optimization at inference time.

Modeling PaSh's Core Transformations

Lukas Lazarek
Northwestern University

Implementing a transformation is not enough — gaining confidence that the implementation is correct is critical too. This talk will be a brief introduction to PL-style modeling in the context of PaSh, the core of what has already been modeled and proved correct in terms of PaSh's transformations, possible extensions and additions to the current model, and how PLT Redex could serve as a useful tool for (at least parts of) the job.

Speaker bio: Lukas is a soon-to-graduate PhD student at Northwestern University working primarily in the broad area of programming languages. He's particularly interested in the pragmatics and practical application of principles and ideas from the field of programming languages.

Inside the Computational Machine Room

Michael Greenberg
Stevens Institute of Technology

The shell language depends on---and enables---an ecosystem of commands. Improvements to the ecosystem are a rising tide that lifts all boats. What makes a 'good' shell tool? Modern capital-S Software (sold as-a-Service or not) doesn't always play nicely in the shell ecosystem... how can we reconcile the shell's long history with modern needs? What should we build, and how?

Speaker Bio: Michael Greenberg is an assistant professor at Stevens Institute of Technology. He received BAs in Computer Science and Egyptology from Brown University (2007) and his PhD in Computer Science from the University of Pennsylvania (2013); previously, he was an assistant professor at Pomona College. His work has ranged from functional-reactive JavaScript (with Shriram Krishnamurthi at Brown) to runtime verification of higher-order programs (with Benjamin Pierce at Penn) to software-defined networking (with Dave Walker at Princeton) to present activities focused on the POSIX shell and executable formalism.

Towards Serverless Shell Scripting

Nikos Pagonas
Brown University / NTUA

The Unix shell is a very useful tool, with vast applications in many different aspects of science and engineering. Shell scripts are prevalent in areas like data science, where researchers need a quick and concise way of processing large amounts of data, as well as combining all kinds of different commands. In an attempt to achieve greater performance without modifying existing code, prior work has been conducted on automatic shell script parallelization, both on single-node and multi-node setups. However, these attempts don't take advantage of serverless computing, an emerging execution model that promises high elasticity and pay-as-you-use pricing. Considering that queries on large amounts of data are often ad-hoc, execution costs can be greatly reduced with serverless scripts, since no charges are incurred when no scripts are running. Also, the elasticity provided by the serverless paradigm can be effectively combined with automatic parallelization, resulting in increased performance and scalability. The purpose of this project is to explore how shell scripts can be parallelized and executed in a

cost-effective and performant manner, according to the serverless model, and integrate the implementation to the PaSh project.

Speaker Bio: Nikos Pagonas is a student of Electrical and Computer Engineering at the National Technical University of Athens (NTUA). Currently, he is a visiting research intern at Brown, working on the PaSh project. His research interests entail parallel and distributed computer systems, as well as serverless computing.

Try: Inspecting command side-effects before modifying systems

Tianyu (Eric) Zhu
Stevens Institute of Technology

In this talk I will dive into the internals of try — a new higher-order command that lets you run another command in a sandbox to inspect its effects before changing your live system. The try essentially contains and records the effects of the command it is applied to, letting users decide whether to merge its effects onto the underlying system — for example, to virtualize complex (and potentially risky) third-party scripts before permanently committing their results. This talk will take a deeper dive into the internals and architecture of the try, including Linux namespaces (via unshare), tracing, and the overlayfs union filesystem.

Speaker Bio: Tianyu (Eric) Zhu, a second year undergraduate computer science major at the Stevens Institute of Technology in Hoboken, NJ. Eric's study mainly focuses on Linux systems and cybersecurity. Eric has worked on various not-for-profit infrastructure projects.

Bash-Compliant PaSh

Seth Sabar
Brown University

POSIX is a standard that most major shell script interpreters including Bash, Zsh, and Dash aim to abide by. However, most of these interpreters support additional features that users often use in their shell scripts. Because the PaSh suite only works on POSIX-compliant scripts, its scope is inherently limited: Asking users to refactor components of their scripts to make them POSIX-compliant for it to work with PaSh is not only impractical but also defeats the purpose of PaSh providing users with a lightweight tool to speed up their scripts.

To increase PaSh's accessibility, I propose to expand PaSh to be compatible with scripts written in Bash — the leading shell script interpreter in Unix and Unix-like environments such as Linux, Mac OS, and Microsoft's WSL. By expanding PaSh's compatibility to Bash, developers who write Bash scripts with non-POSIX features, such as arrays and arithmetic expressions, would be able to run their scripts with PaSh and benefit from the same speed-ups that many of PaSh's current users are already enjoying.

Speaker Bio: Seth Sabar is a senior from Brown studying computer science and math. During his time at Brown, he went from having no experience with computer science to gaining a strong passion for it — finding a particular interest in computer systems. Outside of school, he enjoys playing sports, hanging out with friends, listening to podcasts, and spending time outdoors. This year, he's super excited to work on extending PaSh to be fully compliant with Bash. He can't wait to dive into the Bash source code and create something that will benefit PaSh users around the world.

From Shell Programs to Data Engineering Airflows, Automatically

Seong-Heon Jung
Brown University

Apache Airflow is an open-source workflow management platform for data engineering pipelines representing complex workflows. Airflow programs allow authoring, scheduling, and monitoring workflows programmatically on a variety of SaaS platforms — assuming developers are willing to write their engineering workflows using Airflow. In this talk, I will discuss how to leverage the PaSh compilation infrastructure to automatically transform traditional workflows available as shell scripts into Apache Airflow programs that can run on a variety of cloud platforms. In the process, I will discuss several challenges and possible design decisions.

Speaker Bio: Seong-Heon Jung is an undergraduate student at Brown CS. He enjoys making different programming languages talk to each other. His research interests include compilation, shell pipelines, and programming language design.

Deep Dive: Zero-Touch Out-of-Order Shell Script Execution

Georgios Liargkovas

Brown University / AUEB

Shell-script optimization is a time-consuming, error-prone task. In this talk I will present hs — an out-of-order script execution environment to use modern computational resources better and optimize shell script execution. This runtime optimization, performed in a controlled environment, evaluates command behaviors speculatively. I will present the internals of hs; I will discuss how it addresses challenges in tracing, process isolation, and scheduling; and I will outline possible future directions for out-of-order shell script execution.

Speaker bio: Georgios Liargkovas holds a Bachelor's Degree in Management Science and Technology at Athens University of Economics and Business. He is a research affiliate at Brown University, working on the PaSh project. His research interests span distributed systems, software engineering, programming languages, and machine learning among others.

Discussion: Benchmarks, Workloads, and Evaluation

Nikos Vasilakis
Brown University

This session will focus on several efforts to develop a standard benchmark suite for and around the shell — collecting a set of representative programs for data analysis, orchestration, containment, configuration, and other tasks. The discussion will include the exploration of a conceptual framework — i.e., several dimensions of program characteristics that would be important for such a suite — for picking appropriate benchmarks, the relevant automation infrastructure, as well as updates from and discussion with various PaSh folks — including Dimitra Leventi, Georgios Liargkovas, Seth Sabar, and Seong-Heon Jung.

Speaker bio: Nikos Vasilakis is on the faculty of the Computer Science department at Brown University. His research encompasses systems, programming languages, and security—focusing on automatically enhancing software systems with new capabilities. This enhancement forms a transition path towards secure distributed environments targeting multiple scales. Before joining Brown University, Nikos was a research scientist at MIT. He received his M.Sc. and Ph.D. degrees in Computer and Information Science from the University of Pennsylvania.

---

dCOZ: Distributed Causal Profiling

Yizheng Xie
Brown University

Abstract: Performance analysis is a central concern in modern distributed systems—including ones composed out of black-box software components such as microservice and serverless environments. The increased complexity of these systems complicates their performance analysis, making conventional profilers insufficient for their analysis. In this talk, I will present dCOZ, a causal profiler for distributed programs. dCOZ leverages virtual speedups by rate-limiting communication across the various components — i.e., by effectively slowing down components except the ones that are the focus of an optimization. Performance profiling with dCOZ can identify the correct optimization candidates, quantify their potential end-to-end impact (e.g.,SLOs), and answer what-if scenarios of the form "How will throughput be affected if Service A gets 10% faster?"

Speaker bio: Yizheng is a first-year Ph.D. student at the Brown System Group. Prior to joining Brown, he studied computer science at Boston University and interned in the Kernel Team at Meta. His research interests involve interpretability and fairness in computer systems. He enjoys working on system/software/automation hybrid space either improving the underlying systems or offering system support for various applications.

Verifying an Out-of-Order Shell-Script Scheduler

Siddhartha Prasad & Anirudh Narsipur
Brown University

Reasoning about parallel execution is difficult. In this talk, I will present a lightweight formal model of the scheduling algorithm for speculatively executing shell scripts. I will discuss the guarantees provided and how formal modeling can contribute to system design.

Speaker bio: Anirudh Narsipur is a senior studying Computer Science and Mathematics at Brown University. He is currently pursuing an honors thesis exploring the development of a type system for shell programs. He has previously been a software development intern at the AWS Automated Reasoning Group.

Towards a Type System for Shell Programs

Anirudh Narsipur
Brown University

Robust type systems can help prevent errors,inform optimizations and help developers reason about code. The shell's semantics and ability to run arbitrary commands presents a unique challenge for developing such a system. In this talk, I discuss ongoing work on building towards a type system for shell programs and possible design choices. Our initial focus is on ensuring that sequences of commands are well-typed. We hope to incorporate notions of side effects especially with respect to the file system.

Speaker bio: Anirudh Narsipur is a senior studying Computer Science and Mathematics at Brown University. He is currently pursuing an honors thesis exploring the development of a type system for shell programs. He has previously been a software development intern at the AWS Automated Reasoning Group.

Optimizing PaSh: Adaptive Parameter Selection in Diverse Computing Environments

Oğuzhan Çölkesen
Brown University

In this presentation, we address the challenge of automatic parameter selection in PaSh, a shell script parallelizer, highlighting instances where suboptimal environmental parameters lead to runtimes inferior to standard Bash. Through benchmark examples, we will identify commonalities and demonstrate the necessity for a refined (and automated) parameter selection strategy. How can we construct comprehensive performance profiles for distinct components across diverse computing environments and inputs? Drawing upon existing literature, we examine models utilizing methods such as profiling, Bayesian Optimization, integer programming, and machine learning. The core of our discussion centers on collaborative insights and perspectives on PaSh and our proposed approach, fostering a dynamic academic dialogue. We will then broadly highlight the potential impact of this project, especially in optimizing the efficiency of complex cloud infrastructures where numerous parameters directly influence performance.

Speaker bio: Oğuzhan is a first-year PhD student at Brown University. He's broadly interested in computer systems with a specific emphasis on problems involving parallel and distributed systems and concurrency. Before joining Brown, he graduated from Davidson College in May 2023 with a degree in Computer Science and Mathematics. His previous research experience involves using integer programming to answer graph pebbling questions and designing a

scalable concurrent priority queue as part of his honors thesis work. He's excited to be a part of the PaSh team!

Fault Tolerance for Distributed Shell Script Execution

Zhicheng Huang
Brown University

This talk focuses on adding fault-tolerance support for distributed shell script execution. It identifies key requirements for a fault tolerant system and then presents an early-stage design sketch that applies core techniques behind `hs`—namely, support for tracing and control of side effects—to a distributed context. Gradually the talk will open up to a discussion around the design and implementation of the distributed fault-tolerant version of PaSh.

Speaker bio: Zhicheng Huang is a first-year Sc.M. of Computer Science student at Brown University. Before Brown, he obtained his B.S. in Computer Science and Cognitive Science from the University of Michigan, Ann Arbor. Zhicheng worked in the intersection between Programming Languages and Human-Computer Intersection while at Michigan, and he is interested in exploring more in Systems and Software Engineering.