

YASH LALA

yashlala.com ◊ github.com/yashlala ◊ [linkedin.com/in/yashlala](https://www.linkedin.com/in/yashlala)

(510)-400-5572 ◊ yash.lala@yale.edu ◊ Palo Alto, CA

EDUCATION

Yale Ph.D. in Computer Science
UCLA B.S. in Computer Science

2023 - Present
GPA: 3.782, 2018 - 2022

PROFESSIONAL EXPERIENCE

SOLAR Lab, UCLA CS Department

Student Researcher

Sept 2021 - Sept 2022

Supervisor: Prof. Harry Xu

- Volunteered during the school year, employed full-time to work on kernel patches over the summer. Focused on developing OS kernel mechanisms to allow for transparent memory disaggregation. Worked heavily with kernel programming and debugging tools, such as QEMU + GDB, serial port debugging, and perf.
- Independently developed patches for the Linux kernel's swap subsystems, with the goal of **merging these changes upstream**. Patchset extends the cpuset controller to allow per-cgroup control of active swap devices. Associated refactoring has positive implications for swap throughput, and makes it easy to manage frontswap-based remote memory systems. Code at github.com/yashlala/canvas-linux.
- Developed a patchset to improve the Linux kernel's physical page allocation latency. The patch reduces tail latencies by refilling the percpu low-order free page lists asynchronously using RCU.
- Profiled swapout latencies for RDMA-based remote memory systems under various workloads and prefetch strategies.

CSSI Program, UCLA CS Department

Tutor Undergrad (TA)

July 2022

- Taught introductory data science to high school students for an intensive summer program. Led 4 hours of discussion section and office hours per day, prepared discussion material and assignments, graded papers, and advised students.

Veritas Technologies LLC

SDE Intern

June 2021 - Sept 2021

- Worked on large-scale data consolidation and backup devices (NetBackup Flex platform).
- Implemented functionality allowing Flex nodes to automatically discover new backup nodes over the datacenter network, then to securely assimilate them into a backup cluster. Primarily worked with Ansible, Docker, and various glue languages.
- Replaced SSH-based inter-node communication protocols with a RESTy HTTP based protocol.
- Added web dashboard for backup cluster management.

Pringle Lab, Stanford Genetics Department

Undergraduate Research Intern

June 2017 - August 2017

- Tested algal species for selective binding to various lectin proteins in order to understand the chemical processes behind coral bleaching. Poster available at yashlala.com/pringle-poster.pdf.
- Developed an image recognition program in Java for use in algal cell haemocytometry.

PROJECTS

SC-DNN_{cc}: A Compiler for Stochastic-Computing Accelerators

May 2022 - June 2022

- Developed a compiler backend that transforms programs written in conventional IRs into forms that can be run on a stochastic-computing based hardware accelerator (stochastic accelerators have unusual probability-based programming semantics, and can be difficult to program). Developed a configurable interpreter to emulate stochastic accelerators with a limited set of primitive stochastic operations.

NDN Multicast

May 2022 - Present

- Worked on extending routing protocols for NDN (Named Data Networks). Extended NLSR (a link-state routing algorithm for NDN) to allow for efficient multicast delivery of NDN Interest packets. Student paper available at yashlala.com/nlsr-poster.pdf.

GRU4RecBE: Session Based Recommendations with Features

March 2021 - June 2021

- Developed session-based recommendation system in **PyTorch** which extends the GRU4REC architecture with rich item features extracted from the pre-trained BERT architecture. Non-attentive model outperforms state-of-the-art session-based models over the benchmark MovieLens 1M and MovieLens 20M datasets. **Paper accepted to AAAI Student track.**

MISCELLANEOUS

Passionate about teaching: led student [seminar series](#), volunteered with [ACM TeachLA](#), and currently volunteering as a docent for the [Computer History Museum](#). I'm also interested in retro computing, and am trained as an operator for the [IBM 1401](#).