Ajay Tatachar

+1 (312) 687-4800 | ajaymt2@illinois.edu | ajaymt.github.io

Education

University of Illinois at Urbana-Champaign

B.S. in Computer Science & Chemistry (May 2022, GPA: 3.95/4.0)

Relevant coursework: Computer Systems Engineering, Computer Architecture, Data Structures, Abstract Linear Algebra, Elementary Organic Chemistry

Experience

UIUC Parallel Programming Lab (June 2020 - Present)

Worked on ParaTreeT (Parallel Tree Toolkit), a framework for implementing tree-based parallel applications.

- Refactored the framework to enable greater flexibility and versatility in tree type/structure and domain decomposition.
- Implemented parallel domain decomposition based on space-filling curves for a Barnes-Hut N-Body simulation written with ParaTreeT.
- Gathered extensive performance data and wrote detailed documentation on the code structure and organization of the project.

CS 125: Introduction to Computer Science (January 2019 - May 2020)

Office Hour Captain *(August 2019 - May 2020)*

- Coordinated office hours for a class of ~900 students and ~100 Course Assistants.
- Held ~10 office hours per week to help students with assignments and answer questions.
- Organized and conducted Course Assistant Training to make office hours more effective and efficient.

Course Developer *(January 2019 - May 2020)*

- Developed jtrace, a JVM native agent that traces program state, and java-complexity, a static code analysis tool that calculates cyclomatic complexity. Used to gather data on student homework submissions.
- Worked on janini, an online Java execution platform designed for educational use.
- Held ~4 office hours per week to help students with assignments and answer questions.

Projects

Silk: Compiled systems programming language (Spring 2020)

- C-like semantics with additional safety guarantees and a sophisticated type system.
- Features parametric polymorphism (generics) and simple, modern syntax.
- Leverages the portability and powerful optimization of the LLVM platform.

Mako: Operating System for 32-bit x86 computers (Summer 2019)

- Supports a Linux-compatible ext2 filesystem, pre-emptive and cooperative multitasking, graphical user interface and much more.
- Developed in approximately six months entirely from scratch.

thorin: Debugger for C programs on Linux and macOS (Spring 2019)

- Traces debuggee using ptrace (linux) or mach ports (macOS) and reads DWARF-formatted debug information.
- Provides GDB-like state-inspection features.
- Implemented in Rust and C.

Golsp: Interpreted, general purpose lisp-like programming language (Fall 2018)

- Implemented entirely in Go and itself.
- Features a standard library that provides file I/O routines and an extensible module system.
- Functional purity and Go runtime make it a good fit for concurrent applications.

trash-detector: 36-hour hackathon project developed with two teammates (Spring 2019)

- Convolutional neural network that detects trash (plastic bottles, crushed cans, etc.) in images.
- Gathered data manually (because of a lack of comprehensive trash datasets) to train network.
- Connected the neural network to a web backend and exposed an API to scan and highlight trash in images.

Skills

Languages: C, C++, Python, Go, Rust, Java, OCaml, JavaScript, x86 Assembly, Verilog
Technologies: Node.js, Flask, Django, jQuery, Git, Linux, Charm++, Cocoa/AppKit

Parallel Programming

- Experience designing and implementing highly scalable parallel applications with the actor-based Charm++ framework.
- Experience implementing parallel N-Body simulations based on the Barnes-Hut algorithm.

OS Development and Systems Programming

- Familiar with the x86 architecture: memory virtualization, context switching mechanisms, interrupt handling etc.
- Experience writing system software for linux and macOS.
- Experience developing a multi-tasking operating system kernel capable of hosting user programs and interfacing with common hardware devices.

Programming Languages

- Experience designing and implementing a general-purpose interpreted programming language from scratch.
- Working with ANTLR and yacc LALR(1) and LR(1) grammars, and implementing a parser by hand.
- Experience designing and implementing a systems programming language and compiler frontend for the LLVM platform.