Gregory Schare

818 454 4044 · gs3072@columbia.edu · github.com/gschare · linkedin.com/in/gregory-schare

EDUCATION

Columbia University, New York, NY May 2024
B.A. in Computer Science and Mathematics. (GPA: 3.83)
Selected coursework: Formal Verification. Analysis of Algorithms. Compilers. Formal Semantics. Systems Programming. Computer Systems. CS Theory. Cryptography. Modern Analysis. Modern Algebra. Honors Math. TA for Compilers.

PROFESSIONAL EXPERIENCE

Research Assistant, SSlang Project, Columbia CS. New York, NY September 2022 – present

- Working on compiler optimization and novel reference-counting algorithms for `sslang`, a functional language for real-time programming based on the Sparse Synchronous Model and implemented in Haskell, with Prof. Stephen Edwards.

Software Engineering Intern, Evolution Team, CertiK. New York, NY June 2022 – September 2022

- Brought 4-year stagnated project into production, allowing auditors to automatically verify Solidity NFT contracts.
- Patched tool to verify 30 contracts for high-profile client while tool development was still in progress.
- Designed improvements to formal specification language, including streamlined syntax, loop and contract invariants.
- Implemented translator which parses Solidity labeled with Hoare triples and produces solvable SMT expressions.

Programmer, Making and Knowing Project, Columbia University. New York, NY June 2020 – June 2022

- Achieved 77x speedup of text analysis pipeline by replacing regex with XML parsing using Python lxml library.
- Rendered data analysis tool more convenient and accessible for non-technical researchers by refactoring 7000 lines of legacy code, implementing category and property filters, and providing 3 additional semantic similarity metrics.
- Led archiving, data cleanup, and fully navigable web presentation of over 1000 student lab reports and essays using Google Drive API and Pandoc, with emphasis on minimal computing principles.
- Generated 2 static sites to exhibit 38 experimental projects and almost 2000 editorial discussions by augmenting Pandoc and Jekyll with custom content management and templating systems written in Haskell and JavaScript.

PROJECTS

Imperative Programming Language. *Programming Languages and Translators* with Professor Ronghui Gu.
As a team of 5 students, designed and implemented an imperative programming language designed to enable type-safe task automation scripting. Prepared language proposal, formal language specifications, grammar, and final language report. Implemented language in OCaml using ocamllex for scanning and ocamlyacc for parsing. Built abstract syntax tree, semantic checker, and compiler to LLVM. Highlights: dynamic Python-like lists, C-like structs, Bash library functions.

Scheme Interpreter. *Structure and Interpretation of Computer Programs.*
Implemented a meta-circular evaluator, i.e. a Scheme interpreter written in Scheme.

Visual Algorithms: Metaballs, Floyd-Steinberg Dithering, Boid Flocking.
Metaballs: algorithm for dynamically blending 2D shapes. Floyd-Steinberg dithering: algorithm for reducing number of colors in images while preserving quality. Boids: simulation of complex bird flocking behavior based on simple rules.

HTTP Web Server. *Advanced Programming* with Professor Jae Woo Lee.
Developed HTTP web server and data server from scratch using Unix sockets API in C. Implements three-tier architecture. In addition to serving static HTML and media, the web server communicates over TCP with data server to dynamically deliver results of searching a database.

Programming Challenge in Haskell. *Advent of Code 2020.*
Solved 25 days of programming challenges. 78% pure functionally programmed in Haskell. Highlights: comonads, functional caching using lazy evaluation of infinite data structures, Chinese Remainder Theorem, parsers and domain-specific language implementations. Solutions available at github.com/gschare/aoc2020.

TALKS

Categorical Construction of Programming Languages, Formal Semantics of Programming Languages, Fall 2022.
Automated Formal Verification of Solidity Smart Contracts, CertiK, Summer 2022.
Social Choice: Noise Stability and Arrow's Theorem, Analysis of Boolean Functions, Summer 2021.

SKILLS

Programming Languages: Python, C, Haskell, OCaml, Java, Racket, MIPS, JavaScript, HTML, CSS.
Technologies: Git, Unix, Node, React, SQL, Processing, Jekyll, AWS, Digital Ocean, Next.js.