Runming Li

Contact: 412-403-9281
runmingl@andrew.cmu.edu
https://www.andrew.cmu.edu/user/runmingl/

Research Interest

Programming Languages, Type Theory, Logic, Formal Verification, Proof Assistants

Education

2020 – 2023 **Carnegie Mellon University**, Pittsburgh, PA, United States
Bachelor of Science in Computer Science
Concentration in *Principles of Programming Languages*
GPA: 3.87/4.0

Paper

Preprint 2023 **Runming Li**, Harrison Grodin, and Robert Harper. 2023. A Verified Cost Analysis of Joinable Red-Black Trees. arXiv:2309.11056 [cs.PL]

VISSOFT 2023 **Runming Li***, Keerthana Gurushankar*, Marijn Heule, and Kristin-Yvonne Rozier. 2023. What's in a Name? Linear Temporal Logic Literally Represents Time Lines. In *Proceedings of the 11th Working Conference on Software Visualization (VISSOFT 2023)*. `https://research.temporallogic.org/papers/LGHR23.pdf`

Research Experience

2023 **A Verified Cost Analysis of Joinable Red-Black Trees** in cost-aware logical framework (calf) in Agda: mechanized a formally verified implementation and cost analysis of the parallel sequence library using joinable red black tree, efficient in parallel bulk operations such as set union, in Agda theorem prover, under cost-aware logic framework; featuring dependent type theory, call-by-push-value paradigm, phase distinctions, and interactive theorem prover

Supervised by Professor Robert Harper

2022 **Linear Temporal Logic Explainability**: designed an algorithm and implemented a visualization tool that illustrate semantic properties of linear temporal logic (LTL) formulas as intuitive timelines, for the purpose of specification validation of the LTL formulas; featuring Büchi automata, ω-regular expressions, and forward-chaining search

Supervised by Professor Marijn Heule and Professor Kristin-Yvonne Rozier (Iowa State University)

2022 **Linearly-typed Intermediate Representation** for Resource Aware ML in OCaml: designed and implemented an intermediate representation to facilitate automated amortized resource analysis; featuring linear/affine type system, let-share normal form, and program translations

Supervised by Professor Jan Hoffmann

Work Experience

2023 **Amazon**, *Software Engineering Intern - Compiler*, Sunnyvale, CA, United States.
Worked on designing and implementing compilers and interpreters for domain-specific languages for orchestration platform

2022 **CertiK**, *Software Engineering Intern - Static Analysis*, New York, NY, United States.
Worked on performing static analysis on blockchain smart contracts in Solidity programming language to automate vulnerability detection

Selected Coursework

Programming Languages 15-312 Programming Language Theory · 15-411 Compilers · 15-414 Automated Program Verification · 15-819 Advanced Programming Language (Ph.D. level)

Logic 15-317 Constructive Logic · 15-816 Automated Reasoning and Satisfiability (Ph.D. level) · 15-836 Substructural Logics (Ph.D. level)

Teaching Experience

Fall 2022
Spring 2023
Fall 2023 **15-312 Foundations of Programming Languages**, *Teaching Assistant*, Carnegie Mellon University.
Junior-senior level programming language theory course, topics include: λ-calculus, inductive and coinductive types, continuations, polymorphism, parallelism, concurrency

Spring 2022 **15-317 Constructive Logic**, *Teaching Assistant*, Carnegie Mellon University.
Junior level logic course, topics include: natural deduction, proofs as programs, sequent calculus, logic programming, modal logic, linear logic, session types

Spring 2021
Fall 2021 **15-150 Principles of Functional Programming**, *Teaching Assistant*, Carnegie Mellon University.
First-year level introduction to functional programming course, topics include: programming in Standard ML, datatypes, higher-order functions, continuation passing style, lazy programming

Spring 2022
Fall 2022
Spring 2023
Fall 2023 **98-317 Hype for Types**, *Instructor*, Carnegie Mellon University.
All level survey course on programming language theory and type theory, topics include: λ-calculus, algebraic datatypes, phantom types, compilers, substructural logics, category theory, dependent types