Feitong Leo Qiao

Contact Information

Phone: +1 (646) 830-6529
Email: flq2101@columbia.edu
GitHub: leoqiao18
Website: leoqiao18.github.io

Research Interests

Program Synthesis/Verification, Reactive Systems, Type Theory, Compilers

Education

Columbia University, School of Engineering and Applied Science — **Fall 2022 - Present**
M.S. in Computer Science — *New York, NY*
GPA: 4.0/4

Columbia University, Columbia College — **Fall 2018 - Spring 2022**
B.A. in Computer Science & B.A. in Mathematics — *New York, NY*
GPA: 3.7/4

Teaching Experience

TA: Parallel Functional Programming — **Fall 2022**
Instructor: Prof. Stephen A. Edwards — *Columbia University*

TA: Fundamentals of Large-Scale Distributed Systems — **Spring 2022**
Instructor: Prof. Roxana Geambasu — *Columbia University*

TA: Advanced Computer Networks — **Fall 2021**
Instructor: Prof. Ethan Katz-Bassett — *Columbia University*

TA: Computer Networks — **Spring 2021**
Instructor: Prof. Henning Schulzrinne — *Columbia University*

Research Experience

"Statically Inferring Usage Bound for Infrastructure as Code" — **Fall 2023 - Present**
with Prof. Mark Santolucito. Working in progress.

(Abstract) Infrastructure as Code (Iac) has enabled cloud customers to have more agility in creating and modifying complex deployments of cloud-provisioned resources. By writing a configuration in IaC languages such as CloudFormation, users can declaratively specify their infrastructure and CloudFormation will handle the creation of the resources. However, understanding the complexity of IaC deployments has emerged as an unsolved issue. In particular, estimating the cost of an IaC deployment requires estimating the future usage and pricing models of every cloud resource in the deployment. Gaining transparency into predicted usage/costs is a leading challenge in cloud management. Existing work either relies on historical usage metrics to predict cost, or on coarse-grain static cost analysis that ignores interactions between resources. Our key insight is that the topology of an IaC deployment imposes constraints on the usage of each resource. We propose a system for fine-grained static cost analysis that works by modeling the inter-resource interactions in an IaC deployment as a set of SMT constraints. This allows customers to have formal guarantees on the bounds of their cloud costs.

Temporal Stream Logic (TSL) — **Fall 2022 - Present**
with Barnard PL Labs led by Prof. Mark Santolucito.

- Contributed to the implementation of the synthesis pipeline for TSL, a temporal logic designed for reactive software synthesis.
- Actively working on new theoretical extensions and synthesis techniques for TSL.

Sparse Synchronous Model (SSM) and Language (SSLANG) — **Fall 2021 - Present**
with research group led by Prof. Stephen A. Edwards and John Hui.

- Contributed to the implementation of *SSLANG*, a real-time functional synchronous programming language with deterministic concurrency.
- Led the type system group and hosted weekly meetings.
- Implemented compiler components, such as constraint-based HM(X) type elaboration, pattern-match anomaly detection, pattern-match compilation, etc.

Causal Tracing from System Logs through Natural Language Processing **Spring 2020**
Undergraduate research project, supervised by Prof. Junfeng Yang.

- Explored application of natural language processing models in system log analysis.
- Used BERT language model to trace root causes of errors for systems like Apache Web Server.

Industry Experience

Amazon **Summer 2022**
SDE Intern - AWS IAM *Seattle, U.S.*

- Designed and implemented a server failure detection and recovery system for the AWS IAM Core Services Team.

Amazon **Summer 2021**
SDE Intern - AWS IAM *Seattle, U.S.*

- Designed and implemented a data propagation system for the AWS IAM Core Services Team.

Nexar Inc. **Summer 2019**
DevOps Engineer Intern *Tel Aviv, Israel*

- Contributed to the migration to a Terraform-managed cloud infrastructure and a new CI/CD pipeline.

Megvii **Summer 2018**
SDE Intern *Beijing, China*

- Contributed to the development of a CNN-based SLAM robot and related software toolsets.

Projects

Pocaml: poor man's OCaml

- A compiler written in OCaml for an OCaml-like functional language, with features such as polymorphic let-in bindings, lambda functions, pattern matching and a small standard library.

Pac-Man clone on custom FPGA graphics

- Implemented custom FPGA circuits for general-purpose hardware-accelerated 2-D sprite-and-tile graphics API.
- Implemented game logic, sprite graphics and game AI in C.
- Implemented drivers for the custom hardware and USB SNES controllers in C.

PM: a parallelized minimax chess engine in Haskell

- A minimax Chess Engine implemented in Haskell with a combination of parallelization strategies and alpha-beta pruning.

Spoof: an IOS stickers app

- An IOS app to create, send, and share iMessage stickers. Available on IOS App Store.

Seminars & Reading Groups

Seminar on Theoretical Computer Science **Fall 2022**
(Co-organizer) Formal Semantics of Programming Languages Group. *Columbia University*

Coq Learning Group **Summer 2022**
Weekly reading group with Columbia students. *Columbia University*

Category Theory for Computer Scientists **Fall 2021**
Weekly reading group with Columbia and Barnard students and professor. *Columbia University*

Additional Information

Programming Languages

- Haskell, OCaml, C, Python, Go, Rust, Coq, JavaScript, Java, Swift, Dart, Nix

Languages

- English (Native), Chinese (Native)

---

last updated: November 21, 2023