Leiqi Ye

LinkedIn | GitHub |

Location: Edinburgh, Scotland
Email: Leiqi.ye@ed.ac.uk

Education

University of Edinburgh — Edinburgh, Scotland
PhD, Computer Science (Expected) | Informatics Graduate School PhD Scholarship — *Sep 2024 – present*

University of Michigan — Ann Arbor, Michigan
Bachelor of Science in Computer Science, GPA 3.85/4.0 — *Sep 2020 – Dec 2023*

Publications

Leiqi Ye, Yixuan Li, Guy Frankel, Jianyi Cheng, Elizabeth Polgreen. Unlocking Hardware Verification with Oracle Guided Synthesis, The 25nd Conference on Formal Methods in Computer-Aided Design (FMCAD), October 2025.

Adwait Godbole, **Leiqi Ye**, Yatin A. Manerkar, Sanjit A. Seshia. Modelling and Verification of Security-Oriented Resource Partitioning Schemes, The 23nd Conference on Formal Methods in Computer-Aided Design (FMCAD), October 2023.

Research Experience

PhD Student — Sep 2024 – Present
University of Edinburgh, School of Informatics — *supervised by Elizabeth Polgreen*

- Conducting research in formal methods with a focus on program synthesis, assertion generation, and hardware model verification
- Working with SMT solvers (CVC5), model checkers (EBMC), and syntax-guided synthesis tools to automate specification generation
- Applying techniques such as bounded model checking, inductive synthesis, and static analysis to Verilog designs and benchmarks

Undergraduate Research Assistant — Apr 2022 – June 2024
University of Michigan, EECS department — *supervised by Yatin A. Manerkar*

- Collaborating with Professor Sanjit Seshia and his research group at the Univeristy of California, Berkeley on the UCLID5 verification tool
- Contributed to formal verification of memory subsystems, including DAWG cache architecture, and co-authored a research paper under review
- Simulated AMD GCN3 GPUs in Gem5 to analyze Rowhammer risks from unmalicious programs and architectural consistency issues

Projects

SMART: Oracle-Guided Specification Mining for Hardware Verification — Source Code

- Developed a SyGuS-based framework to automatically synthesize SystemVerilog Assertions (SVAs) from Verilog designs using simulation traces and counterexamples
- Applied bounded model checking (via EBMC) and SMT solving (via CVC5) to refine candidate assertions with oracle-guided inductive synthesis
- Achieved high mutation detection rates (up to 100%) on ISCAS and GoldMine benchmarks without requiring hand-crafted templates

Unmalicious Programs Triggering Rowhammer on GCN-3 Architecture — Source Code

- Simulated AMD GCN3 GPU architectures in Gem5 and wrote monitoring code to detect unmalicious security problems
- Analyzed Rocm code efficiency compared to CUDA code
- Detected possible Rowhammer security issues due to multilayer consistency module differences in unmalicious benchmark

A 2-way Superscalar R10K Out-of-Order Processor Source Code

- Collaborated with teammates to synthesize and devise a 2-way R10K out-of-order execution processor with System Verilog (VCS). Managed executing stage, reservation station, system integration, and LSQ
- Advanced features: 2 Way Superscalar, LSQ with speculating on load dependencies and forwarding, Hardware Prefetch,4-Way Associative L1 Cache, Branch predictor, Visual Debugger, Automated regression testing

Code Size reduction Compiler Source Code

- Developed an Improved Implementation of FMSA (Function Merging by Sequence Alignment) to Decrease Code Size in LLVM, Achieved by Merging Similar Functions and Optimizing Memory Utilization on Resource-Constrained Platforms
- Achieved a 21% increase in compilation speed compared to the best configuration of the original FMSA, with only a minimal 0.5% decrease in code size reduction

Decaf Compiler Source Code

- Implemented Lexical Analysis with Flex and Regular Expressions for Effective Text Processing
- Conducted Syntax Analysis on Generated Tokens and Constructed Corresponding Abstract Syntax Tree (AST) using Yacc and C++
- Achieved a 1.8x Time Speed-Up in Code Execution by Generating MIPS Code in the Backend, Implementing Chaitin's Register Allocation Algorithm, and Utilizing Control Flow Profile Information for Instruction Hoisting

Accelerating Ultrasound Beamforming on the Xeon Phi Source Code

- Achieved a 100x speedup over the baseline single-threaded performance on the Xeon Phi
- Implemented Project with AVX-512 Instructions and Pthreads (POSIX Threads) for Enhanced Performance and Scalability

Technical Skills

- **Verification Tools:** cvc5, EBMC, CBMC, Uclid5, Murphi
- **Synthesis & Analysis:** SyGuS, Static Analysis, Bounded Model Checking, Property Mining
- **Programming Languages:** C++, Python, SystemVerilog, Scala, OCaml, Java
- **Hardware & Compiler Tools:** Cocotb, Yosys, Bison, Flex
- **High-Performance Computing:** CUDA, Rocm, Intel Intrinsics, SIMD

Academic Service

External Reviewer 2025

TACAS (Tools and Algorithms for the Construction and Analysis of Systems)