Lingjun Liu

I build AI agents, and I test whether they actually work.

lingjunliu.se@gmail.com | +1 9847428522 | lingjunliu.github.io | linkedin | github.com/lingjunliu

EDUCATION

North Carolina State University (NCSU) — Raleigh, NC, USA
*Ph.D. in Computer Science | GPA: **4.0/4.0*** — *Aug 2024 – May 2028 (Expected)*

Korea Advanced Institute of Science and Technology (KAIST) — Daejeon, South Korea
M.S. in Computer Science | GPA: 3.53/4.3 — *Sep 2019 – Aug 2021*
Exchange Program in Computer Science — *Aug 2018 – Jun 2019*

National Tsing Hua University — Hsinchu, Taiwan
B.S. in Computer Science | GPA: 3.75/4.3 — *Sep 2014 – Jun 2019*

SKILLS

Languages: Python, C++, Java, R
Frameworks: Git, Docker, REST API, Java Spring, MongoDB, Advanced Installer, SLURM, Linux, LangChain
CI/CD: Jenkins, GitHub Actions
Testing & Systems: Mutation testing, differential testing, fuzzing, static analysis, SMT solvers (Yices, Z3), CI/regression testing

EXPERIENCE

North Carolina State University — Raleigh, NC, USA
Graduate Research Assistant — *Aug 2024 – Present*

IssueMut: Learning Compiler Fuzzing Mutators from Historical Bugs (MSR 2026)
- Built a Python data pipeline that mined **1,760** historical GCC/LLVM bug reports via GitHub API and HTML parsing into structured test-case pairs, powering IssueMut, a bug-detection framework for C compilers
- Designed and deployed an **agentic LLM workflow** using LangChain and Gemini — a multi-agent pipeline with generation, validation, and iterative-revision agents — to synthesize **587 fuzzing mutators** from mined test cases; found **65 bugs (60 confirmed/fixed)** by GCC/LLVM maintainers
- Integrated mined mutators into two leading mutational fuzzers (MetaMut and Kitten); augmented variants discovered up to **1.9x more unique crashes** than their baselines over 24-hour fuzzing campaigns
- Mentored undergraduate researchers on mutator scripting and compiler bug reporting, accelerating expansion of the mutator library for IssueMut and shortening the time from bug discovery to reporting

Differential Testing for Deep Learning Compilers
- Co-led a differential-testing pipeline for deep-learning compilers, parallelized via SLURM job scheduling to run multiple test campaigns concurrently; found **76 bugs** across PyTorch, JAX, TensorFlow, and XLA, **45 confirmed or fixed**

Suresoft Technologies Inc. — Seongnam, South Korea
Associate Research Engineer — *Nov 2022 – Jul 2023*
- Shipped production C++ rule checkers for AUTOSAR C++14, expanding a **200+ rule suite** as a core member of a **3-person team that later grew to 5-7 engineers**; advanced from new hire to core contributor within **2 months** and increased rule-checker **throughput 3x**, from 1-2 per week to 5 per week
- Built a backend license-validation service with Java Spring that enforced time-limited licensing for commercial customers, ensuring compliance and preventing unauthorized use
- Designed and built a rule-description generator using Java Spring and MongoDB, exposing a REST API for frontend data retrieval and creating the database schema to streamline access to rule metadata for internal tools
- Managed software packaging with Advanced Installer and collaborated with QA to debug packaging-related test failures
- Documented edge cases and rule checking logic in Jira for code review; supported regression testing for web-based product releases

KAIST — Daejeon, South Korea
Full-time Researcher — *Nov 2023 – Apr 2024 | Oct 2021 – Aug 2022*
- Built the core reliability computation engine (Bayesian Belief Network modeling) for nuclear safety-critical software — developed an initial R/WinBUGS/Shiny prototype, then migrated it to PyMC
- Designed calculation logic to determine test counts required to achieve target reliability thresholds based on pass/fail outcomes
- Containerized a full-stack application (Next.js, FastAPI, MySQL) with Docker and Docker Compose, and built a one-command deployment script to automate multi-container builds

Graduate Research Assistant — *Aug 2018 – Aug 2021*
- Developed MuFBDTester, an automated test generator for safety-critical nuclear reactor protection system using mutation testing and SMT constraint solving (Yices); achieved **100% detection** of real industrial faults, cut manual test-suite creation from days to minutes, and ran **20x faster** than leading structural-coverage tools (published, **STVR 2022**)
- Modeled multi-agent attack scenarios and generated test cases via model checking for air-traffic-control-system security testing

AWARDS & FELLOWSHIPS

- University Graduate Fellowship (NCSU), Best Short Paper (KCSE 2024), Best Artifact (SEAMS 2021)