roife

☎ [redacted] | ✉ roifewu@gmail.com | roife | roife.github.io

Education

Nanjing University 2023.09 - 2026.06 (expected)

Master's Degree in *Computer Science and Technology* | PASCAL Lab. Tutor: [redacted] Li | Focus on PL and Program Analysis.
TA: **Principles and Techniques of Compilers** (Spring 2024)

Beihang University 2019.09 - 2023.06

Bachelor's Degree in *Computer Science and Technology* | GPA 3.84/4.00, qualified for postgraduate recommendation.
TA: **Programming in Practice** (Fall 2020), **Object-oriented Design and Construction** (Fall 2021, Spring 2022 | S.T.A.R. team)

Work Experience

NVIDIA 2025.02 - Present

OCG (Optimizing Code Generator) team — GPU Compiler LLVM Backend Intern

- Participating in unifying the memory instruction vectorizer between NVIDIA GPU graphics compiler and NVVM, ensuring the graphics compiler's vectorizer aligns with LLVM upstream:
  - Designed an encoding scheme for **multi-address graphic memory instructions** based on the core algorithm of LLVM's memory vectorizer, implementing support for multiple GPU graphic memory instructions while minimizing divergence from upstream;
  - Added several GPU memory instruction vectorization optimizations, such as support for irregular memory instruction;
  - Contributed to a new pass for inferring memory access instruction offset alignment width, improving vectorizer performance.

Rust Foundation Fellowship Program 2024.09 - 2025.09

Rust Foudantion Fellowship (about 20 people globally) — Project Fellow

- As one of the rust-analyzer (official Rust IDE) **maintainers**, ranked in the **top 1%** of contributors; participated in issues handling, PR reviews, and maintenance work across most project modules:
  - Implemented features like control flow highlighting, snapshot test updates, and participated in numerous bug fixes, enhancing IDE capabilities in code understanding and auto-generation;
  - Wrote a **SIMD** implementation for the unicode line breaking module for ARM NEON, achieving a **6.5x** speedup;
  - **Emergency incident response for v0.3.1992**: 4 hours after release, the community discovered a critical bug causing resource exhaustion. I identified the issue **in 3 hours** and designed a new algorithm as fix. This emergency fix controlled the incident's impact, preventing disruptions for global Rust developers.
- Contributing to other projects in the Rust language community, such as rust-clippy;

Awards

- 2022 **National Scholarship** (ranked 1/195 in the major), **Outstanding Graduate** of Beihang University;
- **First Prize** in the 2021 NSCSCC Compilation System Design Competition (Huawei Bisheng Cup), ranking 2nd overall;
- **First Prize** in the Lanqiao Cup C++ Programming Contest (Beijing Division) and **Third Prize** in the National Finals;
- Additionally awarded over ten provincial and university-level awards and scholarships.

Projects

Vizsla — roife/vizsla (WIP)

Modern IDE for Chip Frontend Design · Master's Thesis Project — Rust / SystemVerilog

- Implemented a **semantic analysis framework** and IDE infra for SV, aiming to equip chip design with modern IDE features;
- Based on an **incremental computation** architecture, designed and implemented an incremental analysis IR and specialized passes for efficient and accurate on-demand analysis;
- Project achieves **industry-leading standards** in functionality, performance, and usability: completed **dozens of** modern IDE features for SystemVerilog including code navigation, semantic refactoring, completion, and diagnostics with **millisecond-level** latency;
- Based on the Language Server Protocol, ensuring compatibility with VS Code, Emacs, NeoVim, and other mainstream editors.

Ailurus — roife/ailurus (WIP)

Experimental Programming Language and Toolchain Design · Personal Interest Project — Rust

- Based on **Martin-Löf type theory**, supporting **dependent types**, dependent pattern matching, and inductive datatypes. Implements propositional equality and uses Normalization by Evaluation for equivalence checking, enabling simple theorem proving;
- Features **typeclass-based ad-hoc polymorphism** with **operator overloading** for flexible code reuse;
- Implemented a **module system** for namespace management and encapsulation, addressing code organization in large projects;

- Serves as an experimental platform to explore collaborative design architectures for modern programming language toolchains (compilers, IDEs), aiming to enhance development efficiency and maintainability.

Ayame No-SF-Work/ayame
Compiler from SysY (C subset) to ARMv7 · Bisheng Cup Competition Project Java / LLVM-IR / ARM

- Collaborative project, primarily responsible for backend optimizations on Machine IR, including **iterative register coalescing**, **instruction scheduling**, dead code elimination, and peephole optimizations. Also contributed to syntax tree visitor;
- Handled project testing and DevOps, setting up workflows with Docker and GitLab CI, and writing Python for automated testing;
- The project, built from scratch, featured a complete compiler pipeline (parsing to code generation) with extensive SSA IR and Machine IR optimizations. It ranked **2nd overall** in the competition, achieving **1st place in nearly half of the testcases** and outperforming `gcc -O3` and `clang -O3` on 1/3 of the examples.

LLVM-Lite roife/llvm-lite
Lightweight Edge-side Compiler for Neural Network Operators · Undergraduate Thesis Project C++ / LLVM-IR

- Aimed to utilize known **shape information** from edge inference devices for **secondary optimization** of deep learning operators, reducing runtime spatial and temporal overhead;
- Included a **lightweight compiler** on inference devices and **trimming** work of the LLVM Codegen module. For target workloads, implemented optimizations such as **SCCP** and **DCE**, minimize overhead while maximizing results;
- Successfully reduced inference time by 6% and binary file size by 38% for convolution and softmax operators; implemented **parse-time optimizations** that reduced compilation time by 60% and memory usage by 60%; received **excellent** evaluation for the thesis.

Open Source Contributions

- **Rust Organization** (rust-analyzer contributors team) member, primarily maintaining rust-lang/rust-analyzer
- Also contributed to rust-lang/rust, rust-lang/rust-clippy, rust-lang/rustup, rust-lang/rust-mode
- llvm/llvm-project, clangd/vscode-clangd, google/autocxx, yuin/goldmark, more projects on GitHub.

Skills

- **Programming Languages**: Not limited to specific language. Especially proficient in C, C++, Rust, Java, Python, JavaScript/TypeScript, Verilog/SystemVerilog. Comfortable with Ruby, Swift, OCaml, Haskell, Coq, Agda, etc.
- **PL Theory**
  - Familiar with formal semantics and theory of computation. Experienced with theorem provers (e.g., Coq, Agda).
  - Knowledge of theory and implementation of **type systems** (e.g., Hindley-Milner, System F, Dependent Types).
- **Compiler Design**: **3 YoE**, proficient in full compiler pipeline development, especially **compiler optimizations**
  - Understanding of implementing multi-paradigm PLs; familiar with implementing PL features like bidirectional type checking.
  - Familiar with various **IRs** (e.g., SSA, MLIR, CPS) and **optimizations** (e.g., Mem2Reg, GVN/GCM, register allocation).
  - Knowledgeable about LLVM and LLVM-IR, including its optimization passes and codegen module.
- **Program Analysis**: Familiar with static analysis algorithms (e.g., dataflow analysis, CFA, IFDS, pointer analysis).
- **IDE Development**: **2 YoE**. Familiar with IDE architecture based on **incremental computation** (esp. rust-analyzer, clangd). Knowledgeable about plugin development for VS Code, Emacs, etc., and proficient with the LSP.
- **Computer Architecture**: Familiar with ARM, x86. Understanding of OoO execution, branch prediction and NVIDIA GPU arch.
- **Development Environment**: Proficient in Emacs; comfortable working in macOS and Linux; skilled in leveraging AI tools.

Misc

- **Club**: Served as President of the Beihang OpenAtom Open Source Club, organizing multiple technical sharing and exchange events;
- **Languages**: Chinese (native), English.