Jakob Hain

jakobeha.github.io | github.com/jakobeha jakobeha@fastmail.com | Winthrop, MA 02152, US

Area of Focus: Programming languages + software engineering

Education

Purdue University - *Computer Science Graduate Program* Sept 2020 - Present

Seeking: PhD in Computer Science

GPA: 3.06 / 4

Notable Classes: CS565 Programming Languages, CS5592TPL Types and PL, CS505 Machine Learning, CS592IML Interpretability of ML

Northeastern University - *Khoury College of Computer Sciences* Sept 2017 - Dec 2019

Degree: Bachelor of Science in Computer Science

GPA: 3.887 / 4. Member of the Honors College

Notable Classes: CS4910 Verified Compilers, CS4620 Building Extensible Systems, CS4500 Software Development, CS4410 Compilers

Publications

Contextual Dispatch for Function Specialization - *OOPSLA 20* Oct 2020

Olivier Flückiger, Guido Chari, Ming-Ho Yee, Jan Ječmen, Jakob Hain, Jan Vitek

R Melts Brains - *DLS 2019* Oct 2019

Olivier Flückiger, Guido Chari, Jan Ječmen, Ming-Ho Yee, Jakob Hain, Jan Vitek

Tools

High performance: Rust, C++, C, Unix / Linux internals

Web development: HTML, CSS, JS/TS, React, esbuild, PostgreSQL

Formal methods: Coq, Haskell

General purpose: Kotlin, Swift/iOS, Java, Python, Lua, Docker, Bash, Git, Excel

Research Experience

NominalScript - *Purdue* Jan 2023 - Present

- Superset of TypeScript with an additional nominal type system, like how Typescript is a superset of JavaScript with a structural type system
- Type system formalized in Coq, language implemented in Rust using tree-sitter to parse

UnderstandableBinary - *Purdue* Sept 2022 - March 2023

- ML to improve readability and disassembly of C/C++ object code
- Fetches and compiles packages from debianstable+vcpkg+conan, decompiles using Ghidra, then fine-tunes a transformer (CodeT5) with the decompiled and original code

Research Experience (cont.)

Ř - *Northeastern PRL* Sept 2017 - May 2020

- R̂ is a JIT compiler for R which uses static analysis and speculation to elide unused reflective data like string variable names, improving performance
- Uses well-known compiler techniques but adapted to handle R's unique evaluation and reflective capabilities: liveness analysis, taint analysis, scope analysis, SSA form, loop peeling, LICM, constant folding, type inference, depot speculation, profiling, and others
- Mainly worked on type inference fixes, serialization, and Software Transactional Memory to "safely" reduce lazy expressions when they don't produce side effects

Teaching Experience

CS307 Software Engineering - *Purdue CS* Sept 2021 - Present

- Course which teaches Industry concepts and ethics, teams create their own software project (e.g. website), and submit design documents, and follow SCRUM
- As project coordinator I help teams specify their projects and review their documents
- As head TA (fall 2022) I also handled logistics and Qs from other coordinators

CS2500 Fundamentals I - *Northeastern CCIS* Sept 2018 - Dec 2018

- Northeastern's mandatory introductory course, teaches foundations of programming (e.g. recursion) and good practices (documentation, testing) in a dialect of Scheme

Work Experience

Intern Developer - *NextDroid (self-driving ground-truth analysis via LIDAR)* Jan 2020

- Fixed website bugs and create camera view for analysis (frontend)
- Fixed camera C++ driver and server (backend)

Freelance Developer - *RemoG, Remote* Jan 2018

- Built an iOS app to show sensor data (speed, temperature, pressure) for a car

Community Service

Counselor - *GER^{2}I, West Lafayette IN* July 2022

Counselor - *Parks and Recreation, Winthrop MA* June 2016 - Aug 2016

Instructor - *Cervizzi's Martial Arts Academy, Winthrop MA* April 2015, Oct 2016

Personal Projects

cge-ai - *General-purpose ML/AI library for turn-based games* Jan 2022 - May 2022

- Based on AlphaZero, but modified to support more flexible games (e.g. more players)

TreeScript - *DSL to transform (refactor) syntax* Dec 2018 - May 2019

- Related: coccinelle (Muller, Lawall, Andersen, Brunel, Hansen, Padiolaeu, Palix),
  "Parser Parser Combinators" (van Tonder, Le Goues)
- Language-agnostic AST match and substitution. Syntax example: "foo(\x) -> bar(\x)"
- Finalist at Northeastern's RISE 2019

Personal Projects (cont.)

Descript - *Simple language which transforms its own code* April 2018 - Aug 2018

- Descript programs can input and output Descript source code, like macros. Unlike macros, Descript programs will modify the input file in-place/on-disk
- Idea: complex large-scale refactors. Evolved into TreeScript to support other languages
- IDE (VS Code) extension which highlights errors and renames symbols

Hobbies: Running, Weightlifting, Graphic Design, Electronic Music Production