

Caner Derici, PhD

[🔗 dericilab.live](https://dericilab.live) [✉️ caner@dericilab.live](mailto:caner@dericilab.live) [🔗 canerderici](https://www.linkedin.com/in/canderderici) [🔗 cderici](https://orcid.org/0000-0002-1349-832X) [📍 UT, USA](#) [✉️ US Citizen](#)

[↗️ extended version](#)

Technical Skills

| | |
|--------------------------------------|--|
| Areas of Expertise: | Compilers & Programming Languages · Machine Learning · Distributed Systems |
| Languages: | C++ · LLVM · MLIR · Go · Python · Racket/Scheme · Java · SQL · JavaScript |
| Workflows & Productivity: | Linux · Neovim · tmux · VSCode · Copilot · Git · GH Actions · Obsidian · Toggl · Todoist |
| API, DB & Misc: | REST · gRPC · OpenAPI · FastAPI · DQLite · MongoDB · PostgreSQL · CI/CD · Jenkins |

Education

| | |
|--|-------------|
| Ph.D., Indiana University, Bloomington , Computer Science, Compilers & Programming Languages | 2015 – 2025 |
| Dissertation: Self-Hosting Functional Programming Languages on Meta-Tracing JIT Compilers | |
| M.Sc., Boğaziçi University , Computer Science, Machine Learning | 2012 – 2014 |
| B.Sc., Bilgi University , Computer Science | 2005 – 2010 |

Selected Projects

[Pycket: A tracing JIT compiler for full-scale bootstrapping functional language](#)

Built the first tracing JIT capable of running a self-hosting language on a meta-tracing JIT backend, and worked on it [for over five years](#) during PhD. Created a new IR and implemented [performance analysis tools](#), run-time optimizations, and [formalisms](#) to improve throughput. Added code-gen for the FFI layer, engines, and meta-continuations for user threads. Used Python, C++, and Racket.

[Around the World in 26 Languages](#)

Building a personal collection of experimental compilers, one for each letter of the alphabet, named after a world city. Each explores different ideas in language design, IR construction, optimizations, and run-time. Targeted backends include LLVM IR, MLIR, NVPTX for CUDA. All in C++.

[Racket IR: Linklets](#)

Contributed to Linklets, the intermediate representation and compilation unit format used in current Racket releases on Chez (see publications). Documented Linklets' design and developed its formal semantics in my [PhD dissertation](#).

[Rax: A full-stack nanopass compiler from Racket to x86_64](#)

Implemented all the passes (closure conversion, register allocation, code-gen, etc.), along with garbage collection. Developed optimizations, such as inlining, loop-invariant code motion, and proper tail calls. In Racket, and C++.

[FARS: Functional Automated Reasoning System](#)

A resolution/refutation theorem prover, for expressions in first-order predicate logic with equality. Used binary paramodulation, and forward and backward subsumption for equational deduction. In Racket.

[HazirCevap \(Witty\): A closed-domain question answering system for high school students](#)

Government-funded large-scale question-answering system. M.Sc. thesis on NLP. Led the R&D team (3 faculties, 4 grad students). Developed a Hidden Markov Random Field model for question analysis, and relevance metrics for information retrieval and response generation (see publications). In Python, and JavaScript.

Experience

| | |
|---|-------------|
| Canonical USA | REMOTE, US |
| SWE II (L4 - IC3) · Domain Specific Language Compiler & Distributed Orchestration | 2021 – 2024 |
| <ul style="list-style-type: none">Designed and developed a custom scripting DSL for (CPU & memory) restricted computations. Built compiler on top of a modified Lua interpreter in Go.Developed and maintained Juju, an eventually consistent distributed orchestration system used by ~200 companies globally, capable of handling 1000+-node workloads with 99.9% availability on any infrastructure (Kubernetes or otherwise) across various cloud providers (e.g., AWS, GCE). All in Go.Improved reliability and fault tolerance by implementing edge machine services on relational DQLite back-end, migrating from NoSQL MongoDB (e.g., sample PR).Owned client libraries for three years—python-libjuju, Terraform Juju Provider; doubled active users and maintained a steady release cadence.Took part in roadmap planning, coordinated cross-team work; mentored junior engineers, and improved hiring by creating a structured bootcamp process that cut down onboarding new engineers from 6 months to 1 month. | |
| Indiana University | IN, US |
| Research Assistant, Course Instructor | 2015 – 2021 |
| <ul style="list-style-type: none">Independently took an ambiguous, uncharted compiler problem to a working product; built the first-ever tracing JIT compiler that is a full-scale runtime for a self-hosting, production-grade language. Conducted a full performance investigation and designed new optimization algorithms (see PhD dissertation).Taught data structures & algorithms, compilers, virtual machines, and domain specific languages. | |

Asseco SEE Group

Software Engineer

2010-2012

- International software company developing virtual payment systems for e-commerce platforms. I developed and delivered 3 virtual point-of-sale projects in 2 years. Used Java, Apache Tomcat, Spring, Mercurial, Jira.
-

Selected Publications

- Flatt M., Derici C. Dybvig R. K., Keep A. et. al. "Rebuilding racket on chez scheme (experience report)", ICFP'19
 - Derici C. et. al. "A closed-domain question answering framework using reliable resources to assist students" Natural Language Engineering'18
 - Derici C. et. al. "Question analysis for a closed domain question answering system", CICLING'15
 - Derici C. et. al. "Rule-based focus extraction in Turkish question answering systems", SIU'14
 - Başar R. E., Derici C., and Şenol Ç. "World With Web: A compiler from world applications to JavaScript". Technical Report, Scheme and Functional Programming Workshop'09
-

Awards & Scholarships

- Scholarship and award for a project on teaching natural languages to hearing impaired, 2014.
 - Full Scholarship for PhD, 2015-2025
 - Full Scholarship for MSc, 2012
 - Full Scholarship for BSc, 2005-2010
-