My research is at boundaries between programming languages, system programming, and computer architectures, with the overall goal of producing systems which are better-understood, more robust, and more secure.

MAIN RESEARCH PROJECTS

On Programming Language Design and Implementation

- Reverse engineered and formalised the Julia subtyping relation, relied upon by Julia runtime to implement multiple dispatch. Built a reference implementation of the algorithm, used for differential testing of Julia runtime.
- Designed LikeTypes, a gradual type system for dynamic languages. This design captures common programmer mistakes, is compatible with object-orientation and can be implemented efficiently. It is implemented in the Thorn language and in the StrongScript dialect of JavaScript.

On Weak Memory Models

- Defined the de-facto standard x86-TSO memory model of the x86 architecture, now relied upon by language standards and mainstream compiler implementations.
- Studied verified compilation for a concurrent dialect of C of top of x86-TSO, building the CompCertTSO verified compiler.
- Studied the correctness of compiler optimisations in the C and C++ memory models. Identified unexpected issues in the C++11 standard, and implemented a tool, cmmtest, to perform random testing of compiler optimizers.

On Tool Support for Semantics

- Designed and implemented the Ott tool for expressing semantics of large programming language definitions.

On Debug Information

- Designed and implemented tools to validate and synthesize DWARF unwind tables from binaries, and to speedup ~20x DWARF-based unwinding.