Zijie Zhou

zhou.zijie@northeastern.edu — https://zijiezhou.me

—

Education

Northeastern University, Boston, MA
M.S. in Software Engineering Systems, 2025

Arizona State University, Tempe, AZ
B.S. in Computer Science, 2022

—

Research Experience

Research Assistant, Number Project (with Prof. Robin Hillyard), 2024–present

- Extended a Scala-based numerical system into a mathematically principled probabilistic domain. Introduced a valuation monad based on Scott-continuous probabilistic valuations, enabling partial distributions and directed suprema for domain-theoretic approximation.
- Contributed to the project's research codebase (`https://github.com/rchillyard/Number`), integrating domain-theoretic semantics into the system's implementation.
- Integrated algebraic structures from Typelevel Cats (Semigroup, Monoid, Group, Ring, Field, Order/PartialOrder). Designed property-based algebraic law tests using Cats Laws + Discipline.
- Defined information ordering and continuous probabilistic bind, turning Gaussian/box fuzzy numbers into extremal cases of a unified semantic space.
- Abstracted the codebase into a core expression language with a compositional denotational semantics, relating exact, interval, fuzzy, and symbolic layers.

Open-Source Contributor, Typelevel Ecosystem (Cats / Monocle), 2024–present

- Working on a principled algebraic foundation for Optics by using Profunctor as the unifying abstraction.

- Designing a Scala 3–based rule-checking and counterexample-generation framework for systematic law testing of Lens, Prism, Iso, and other optics.
- Aiming to provide an "optics-laws" infrastructure analogous to `cats-laws`, strengthening soundness of the ecosystem.

Creator and Author, Well-Typed Project (`https://github.com/Zijie000/well-typed`), 2024–present

- Implemented core calculi from TAPL in OCaml, including STLC, simply typed λ-calculus with references, subtyping, and small-step/big-step operational semantics.
- Mechanized metatheoretic properties (e.g., progress/preservation tests) and built interpreters and type checkers for each calculus.
- Designed the project as a preparation for research in type theory and verified language design, translating formal semantics into executable artifacts.

—

Relevant Coursework and Readings

Readings:

- Steve Awodey, *Category Theory*
- Benjamin C. Pierce, *Types and Programming Languages*
- Robert Harper, *Practical Foundations for Programming Languages*
- Glynn Winskel, *The Formal Semantics of Programming Languages*
- Noel Welsh and Dave Gurnell, *Scala with Cats*

Coursework:

- Big Data Systems Engineering with Scala (2025)
- Enterprise Software Design (2025)
- Programming Languages (Dan Grossman, Coursera), 2024
- Principles of Programming Languages (2021)

—

Honors and Awards

- Successful Participant, Mathematical Contest in Modeling (MCM), 2021

—

Research Interests

Programming Languages, Type Theory, Category Theory, Functional Programming, Homotopy Type Theory.
Focus on:

- Type-theoretic foundations for modular metatheory (logical relations, normalization, and semantic type soundness)
- Algebraic abstractions for numeric computing and error analysis
- Categorical and domain-theoretic semantics for probabilistic or uncertain computation
- Strengthening the Typelevel ecosystem through principled law-testing and semantic foundations