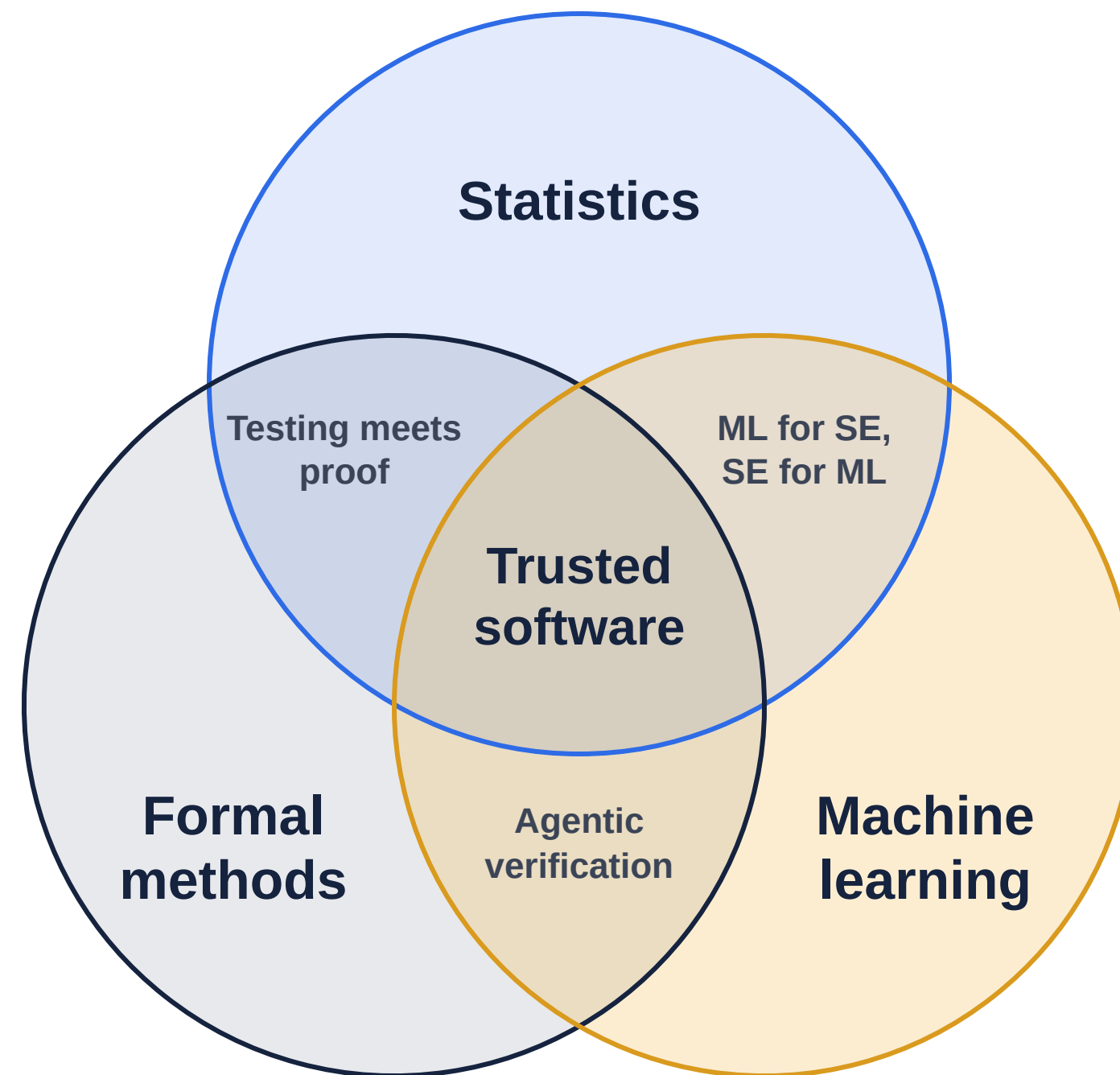


Software we can trust, by design and by evidence

THE QUESTION

When can we trust software that no one can fully understand?



OUR APPROACH

We combine statistical evidence, formal proof, and learning, so that every claim of correctness comes with a measure of confidence.

Research Directions

We study problems like this...



Specification

Can we capture intent as precise specifications that drive development, testing, and verification?



Testing

Can we generate inputs that find bugs, and measure how much risk remains when testing stops?



Verification

Can we automatically prove that a program meets its specification?



Debugging & Repair

Can we trace a failure back to its root cause through program dependencies, and fix it?

...using techniques like this.



Statistical & Causal Inference

Using statistical estimation and causal inference to quantify unseen behavior and program dependence



Formal Proof

Using theorem provers such as Lean to build machine-checked proofs



Machine Learning & LLM Agents

Using learned models and LLM agents to automate test generation and proof construction



Optimization

Using search-based optimization to find better estimators and program configurations

Current Research Topics

Statistical Software Assurance

Quantifying What Testing Has Not Seen

- Estimating residual risk after testing, so that we know when testing is enough (ESEC/FSE'23, ICSE'24, ICSE'25, ICSE'26)
- Building efficient, accurate estimators, so that assurance scales to real software (ICLR'25, ASE'26)

Agentic Verification

Proof and Test Synthesis with Agents

- Guiding LLM agents with property templates to write Lean proofs and property-based tests with fewer hallucinations (ICSE'27)
- Generating structured test inputs and regression tests with LLMs (S&P'26, FSE'26)

Spec-Driven Development

Keeping Spec, Code, and Proof/Test in Sync

- Deriving code and its proofs/tests from a single specification (ICSE'27)
- Tracing changes across spec, code, and proof/test, so that all three stay consistent (FSE'26)

Software under Uncertainty

Assuring Uncertain Systems

- Trusting ML, both inside software and in the tools that build it (ICST'19, FSE'26)
- Telling real bugs from measurement noise in quantum programs
- Quantifying field-failure risk of cyber-physical and robotic software

Application

Who we hope to work with



Self-driven curiosity

Research has no answer key. We look for people who chase their own questions instead of waiting for the next assignment.



Generosity and openness

Hard problems are easier to tackle together. We value people who help each other and stay open to being wrong.



Depth over count

We would rather produce one result that truly matters than many that do not.

How to get in touch



Email me
(seongminlee@sigsoft.org) with:

- Your curriculum vitae (CV)
- Your academic transcript
- Your grad school application form
- Your GitHub profile (if available)



For more information

Visit my homepage nimgnoeseel.github.io