

Siddharth Bhat

Expertise

Formal Verification (Lean) & Static Analysis (LLVM/MLIR)

- Given >4 talks @ LLVM dev meeting (Formal Semantics for LLVM & MLIR).
- Led the `lean-mlir` project for verifying optimizations in MLIR.
- Top 20 contributors to the Lean theorem prover, #2 contributor to Lean's bitvector theory.
- Coauthor of the Lean4 metaprogramming book.

Algorithms (SMT Solving for QF_BV)

- Published new, formally verified decision procedures for parametric bitvector theory.
- Fixed various bugs in Polly's implementation of polyhedral algorithms.
- Designed and implemented algorithms for proving memory (non-)interference in symbolic simulators for ARM.

High Performance Computing (Loop Optimization) & Compilers (LLVM, MLIR, Haskell, Lean):

- #3 Contributor to Polly, LLVM's polyhedral loop optimizer. 121 commits to Polly, 6000 LoC to LLVM, added support for polyhedral GPU code generation.
- Google Summer of Code Mentor for LLVM (2016).
- #2 contributor to Asterius, a Haskell → WebAssembly compiler.
- I implemented a Haskell-style runtime on top of the WASM runtime. Contributions merged into Asterius, and eventually into GHC proper.
- Primary author of Lean's LLVM backend.
- Papers on designing MLIR based IRs for quantum & functional compilation.

AI for Maths (Lean, F*):

- RL for proofs in the F* proof assistant @ Microsoft Research ("Towards neural synthesis for SMT-assisted proof-oriented programming" @ ICSE 2025, best paper).
- Established stronger baselines for AI based geometric theorem proving.

Education

PhD in Comp. Science **University of Cambridge.**

(2024 - 2026 March (tentative))

PhD in Comp. Science **University of Edinburgh (moved to Cambridge).**

(2022 - 2024)

Masters in Comp. Science **International Institute of Information Technology Hyderabad India.**

(2020 - 2021)

B.Tech in Comp. Science **International Institute of Information Technology Hyderabad India.**

2015 - 2020

Internships

Sep-Nov '24 **Amazon Web Services, Automated Reasoning Group, Austin**, Research Intern.

Deciding memory (non)interference in `lnsym`, a Lean-based ARM symbolic simulator

Jul-Sep '23 **Microsoft Research, Redmond**, Research Intern.

Retrieval Augmented theorem proving for the Fstar proof assistant.

May-Jul '19 **Tweag.io, Paris, France**, Research Intern.

Re-implemented portions of GHC (Glasgow Haskell Compiler) runtime for [Asterius \(link\)](#), a Haskell to WebAssembly compiler. Involved Haskell, C, and WebAssembly.

- Summer 2018 **ETH Zurich, Zurich, Switzerland**, Research Intern.
Investigating formal verification of polyhedral compilation. [PolyIR \(Link\)](#) is a formal specification of polyhedral programs.
- Summer 2018 **Google Summer of Code, Polly Labs**, Mentor.
Mentoring a project to enable Polly's loop optimisations into Chapel.
- Mar-Dec '17 **ETH Zurich, Scalable Parallel Computing Lab, Zurich, Switzerland**, Research Intern.
Worked on GPU code generation in Polly, a polyhedral loop optimizer for LLVM.
- May-Jul '16 **Research Intern, IISC Bangalore**, Bangalore.
Worked on PolyMage, DSL compiler for optimising loop transforms. Contributed to ISL and PLUTO. Implemented tiling patterns, optimised PolyMage for stencils.
- Summer 2015 **Google Summer of Code 2015, Google**.
Worked on VisPy, a pure Python graphics library which uses OpenGL internally for performance. Successfully completed.

Publications

First Author Papers

- Certified Decision Procedures for Width-Independent Bitvector Predicates: **Siddharth Bhat (1st)**, Léo Stefanescu, Chris Hughes, Tobias Grosser. OOPSLA 2025
- Verifying Peephole Rewriting in SSA Compiler IRs: **Siddharth Bhat (1st)**, Alex Keizer, Chris Hughes, Andres Goens, Tobias Grosser. ITP 2024
- Lambda the Ultimate SSA: **Siddharth Bhat (1st)**, Tobias Grosser. CGO 2022
- Word Embeddings as Tuples of Feature Probabilities: **Siddharth Bhat (1st)**, Alok Debnath, Souvik Banerjee, Manish Shrivastava Representation Learning for NLP, 2020

Collaborations

- Interactive Bit Vector Reasoning using Verified Bitblasting: Henrik Böving, *Siddharth Bhat*, Alex Keizer, Luisa Cicolini, Leon Frenot, Abdalrhman Mohamed, Léo Stefanescu, Harun Khan, Josh Clune, Clark Barrett, Tobias Grosser. OOPSLA 2025
- Verifying Wu's Method can Boost Symbolic AI to Rival Silver Medalists and Alpha-Geometry to Outperform Gold Medalists at IMO Geometry: Shiven Sinha, Ameya Prabhu, Ponnuram Kumaraguru, *Siddharth Bhat*, Matthias Bethge. NeurIPS 2024 Workshop MATH-AI
- Towards Neural Synthesis for SMT-Assisted Proof-Oriented Programming: Saikat Chakraborty, Gabriel Ebner, *Siddharth Bhat*, Sarah Fakhoury, Sakina Fatima, Shuvendu Lahiri, Nikhil Swamy. ICSE 2024
- Rewriting Optimization Problems into Disciplined Convex Programming Form: Ramon Fernandez Mir, *Siddharth Bhat*, Andres Goens, Tobias Grosser. CICM 2024
- Guided Equality Saturation: Thomas Koehler, Andres Goens, *Siddharth Bhat*, Tobias Grosser, Phil Trinder, Michel Steuwer. POPL 2024
- QSSA: An SSA based IR for Quantum Computing: Anurudh Peduri, *Siddharth Bhat*, Tobias Grosser. CC 2021
- Optimizing Geometric Multigrid Computation using a DSL Approach: Vinay Vasista, Kumudha KN, *Siddharth Bhat*, Uday Bondhugula. Supercomputing (SC), Nov 2017

Open Source Contributions

- Lean4** Co-developed the bitblasting theory for Lean's bitvector automation, wrote the LLVM backend for the compiler.
- Rocq** Submitted issues, bug-fixes, helped improve developer documentation.
- VE-LLVM** Collaboration with VE-LLVM, a formal semantics of the LLVM compiler toolchain in Coq

- Polly** Implementing support for Fortran, added unified memory abilities to the CUDA backend within Polly, a polyhedral loop optimiser for LLVM. ([Link to commits](#))
 - Symengine.hs** GSoC 2016. Haskell bindings to SymEngine, a C++ symbolic manipulation library.
 - VisPy** GSoC 2015. Rewrote scene graph for performance. Added visuals, high level API for easy use of plotting. Implemented auto-resizing with **Cassowary**, a linear optimisation library.
 - Rust** Contributed to the Rust compiler and ecosystem. Found compiler errors, fixed libraries. Was part of **Piston**, group of Rust programmers who experimented with writing game engines.
 - Haskell** Contributed to the Haskell ecosystem. Reported and fixed bugs in *stack*, *stackage*, *diagrams*, *GHC*, etc. ([Link to GHC commits](#)).
 - PLUTO** Source to Source C optimiser for loop nests. Improved the PLUTO API that had gone out of sync with master. Discovered bugs in PLUTO for diamond tiling transforms
 - PolyMage** DSL compiler that generates C code. Uses **Polyhedral Compilation** Extended the compiler to add stencils, time iterated-stencils.
 - PPSSPP** C++ open source PSP emulator. Wrote most of the touch handling code. Implemented atomic locks for audio performance.
- Personal Projects (Compilers)**
- Lean-MLIR** Formal semantics for the MLIR compiler framework, defined within the Lean4 proof assistant.
 - Iz** An MLIR based compiler backend for the Lean4 proof assistant.
 - Simplexhc** A custom compiler for a subset of Haskell. The goal is to try and apply *polyhedral compilation* ideas to compile a lazy, pure, functional programming language with LLVM as a backend. Has **64 stars** on github.
- Personal Projects (Formal Verification)**
- Lean4 Metaprogramming Book** A textbook on metaprogramming in Lean4. I wrote the chapters on tactics and metaprogramming for embedded DSLs.
- Talks & Presentations**
- Barvinok** Talk at ETH Zurich: Slides describing the Barvinok algorithm to count lattice points in polyhedra
 - FunctionalConf '19** Talk on implementing embedded probabilistic programming languages in Haskell ([Slides](#))
 - Haskell Exchange 2020** Talk on optimizing `smallpt-hs` (a port of a raytracer to haskell) to beat C++ performance ([Slides](#))
 - Euro LLVM Dev 2025** [How to trust your peephole rewrites: automatically verifying them for arbitrary width!](#)
 - US LLVM Dev 2024** [lean-mlir: A workbench for formally verifying peephole optimizations in MLIR.](#)
 - US LLVM Dev 2023** [\(Correctly\) Extending dominance to MLIR Regions.](#)
 - US LLVM Dev 2023** [MLIR Side Effect Modelling.](#)
 - Euro LLVM Dev 2022** [MLIR for Functional Programming.](#)
 - FPIIndia 2021** [Equality Saturation.](#)
 - Functiona Conf 2019** [Monad-bayes: Probabilistic programming in Haskell.](#)

Awards

- Renaissance Philanthropy, AI for Maths** 'Towards evaluating Natural Language Profs' One of 30 research groups awarded out of over 280 applicants.