Mike (Deyuan) He

mikehe@princeton.edu · @Mike He cs.princeton.edu/~dh7120

Education

Princeton University, Princeton, NJ 2022 – Est. 2027

Ph.D. in Computer Science
M.A. in Computer Science (Sep 2022 – May 2024)
Advisor: Prof. Aarti Gupta
Fields of study: Compilers; Formal Verification; Distributed Systems; Equality Saturation

University of Washington, Seattle, WA 2018 – 2022

B.S. in Computer Science, GPA: 3.89/4.0 (Cum Laude)
Advisors: Prof. Zachary Tatlock & Dr. Steven Lyubomirsky
Selected Honor: CRA Outstanding Undergraduate Researcher Award, Honorable Mention (2022)

Research

PInfer: Invariant inference for distributed systems from traces Sep. 2023 – Now

Implementing specification miner for distributed systems using production/model traces.

- Implement a syntax-guided synthesis-based invariant generation algorithm
- Formulate invariant synthesis as boolean function learning problems

CatsTail: Synthesizing Packet Programs via Equality Saturation June. 2023 – Jan. 2024

CatsTail is an equality saturation-based P4 program synthesizer. Previous works use **SKETCH** to synthesize the program, which takes too long to make debugging on actual hardware possible. Compared with SKETCH, CatsTail is up to **30x/2000x** faster (preliminary) in finding the optimal stage allocation for Intel Tofino/Domino (Banzai ALU). I lead the design and implementation of CatsTail.

Verifiying correctness of SW/HW mappings June. 2023 – Now

hex is a language for accelerator operation explication and a tool for verifying the software-hardware mapping correctness. My contributions and work in progress are

- Implemented a case study for FlexASR pooling instructions in hex and verified its correctness against the software implementations.
- Designing memory layout mapping invariant inference/generation algorithm.
- Co-authored a paper in submission at **OOPSLA'24**

Improving Term Extraction with Acyclic Constraints Sep. 2022 – Feb. 2023

To have a better term extraction algorithm for egg, an equality saturation framework, we devise the encoding using Weighted partial MaxSAT and include a set of *Acyclic constraints* that ensures the acyclicity of the extracted term. Our encoding demonstrates better solver time (∼3x speed up) for the case study of extracting tensor programs. I led the development of the case study and the encoding, and authored the workshop paper at **PLDI EGRAPHS'23**.

Py*rope*: Towards Correct-by-construction Hardware Modeling Mar. 2022 – June. 2022

Py*rope* is a Python-based framework for high-level hardware modeling. **Py*rope*** enables expressing proofs and guarantees of modeled instruction in Python and transpiles hardware models into Dafny for verification. **I led the development of Py*rope* during my internship at Intel Labs.**

3LA: Application-level Validation of Accelerator Designs June. 2021 – June. 2022

3LA is a software/hardware co-verification methodology for DL accelerators that aids hardware developers in performing early-stage application-level debugging. My contributions are

- Led the development of *flexmatch* and extended Glenside to support a more diverse set of models.
- Implemented the compilation pipeline for VTA using BYOC interfaces of TVM.
- Implemented handwritten digit recognition (on CIFAR) and image classification (on ImageNet) for VTA. Passed the mapping validation using 3LA.
- Co-authored a **ASPLOS LATTE'21** workshop paper.
- Co-authored a paper published at **ACM TODAES**.

Dynamic Tensor Rematerialization Jan. 2020 – Oct. 2020

Dynamic Tensor Materialization (DTR) is an online, heuristic-based checkpointing algorithm that enables DL inference under constrained memory budgets. My contributions are

- Identified problems in the PyTorch DTR implementation.
- Designed the evaluation framework for DTR and extended the case studies to multiple new DL applications (e.g. Unrolled GAN, UNet).
- Co-authored the paper published at **ICLR'21**.

Publications

- Akash Gaonkar, **Mike He**, Yi Li, Bo-Yuan Huang, Andrew Cheung, Vishal Canumalla, Gus Smith, Zachary Tatlock, Sharad Malik, and Aarti Gupta. Verification of software-to-hardware mappings for machine learning accelerators. 2024 [***in submission***]
- **Mike He**, Haichen Dong, Sharad Malik, and Aarti Gupta. Improving term extraction with acyclic constraints. In *E-Graph Research, Applications, Practices, and Human-factors Symposium* (*EGRAPHS'23*), 2023 [Paper]
- Bo-Yuan Huang*, Steven Lyubomirsky*, Yi Li, **Mike He**, Thierry Tambe, Gus Henry Smith, Akash Gaonkar, Vishal Canumalla, Gu-Yeon Wei, Aarti Gupta, Sharad Malik, and Zachary Tatlock. Application-level validation of accelerator designs using a formal software/hardware interface. *ACM Trans. Des. Autom. Electron. Syst.*, 2022 [Paper]
- Bo-Yuan Huang*, Steven Lyubomirsky*, Thierry Tambe*, Yi Li, **Mike He**, Gus Smith, Gu-Yeon Wei, Aarti Gupta, Sharad Malik, and Zachary Tatlock. From dsls to accelerator-rich platform implementations: Addressing the mapping gap. In *Workshop on Languages, Tools, and Techniques for Accelerator Design* (*LATTE'21*), 2021 [Paper]
- Marisa Kirisame*, Steven Lyubomirsky*, Altan Haan*, Jennifer Brennan, **Mike He**, Jared Roesch, Tianqi Chen, and Zachary Tatlock. Dynamic tensor rematerialization. In *International Conference on Learning Representations* (*ICLR'21*), 2021 [ArXiv]

Service

- Member of Artifact Evaluation Committee of PLDI'24, POPL'24, MLSys'23, MICRO'21
- Mentor in the Ph.D. application mentoring program (Princeton, 2023)

</> Internships

Amazon Web Services, Santa Clara, CA May. 2024 – Aug. 2024

Applied Scientist Intern (Formal Methods/**LLMs**)

Developing and refining *PInfer* that mines likely invariants for distributed systems from traces.

- Implementing **syntax-guided synthesis** of likely invariant
- Designing the mining algorithm as boolean function synthesis using **SAT solvers**

Taichi Graphics, Remote and Beijing, China June. 2022 – Sep. 2022

Compiler R&D Intern (**C++**/**Python**)

- Refactored the intermediate representation (IR) of Taichi Language
- Implemented standalone **Tensor type** for better compilation speed

- Adapted **compiler passes** (e.g. Load/Store forwarding, Dead code elimination, reaching definition, etc.) to optimize for tensor type expressions
- Implemented **LLVM**-based code generation for tensor type for Superword-level vectorization

Intel Labs, Hillsboro, OR — Mar. 2022 – June. 2022

Formal Verification Research Intern (Formal Methods/**Python**/**Dafny**)

Developed the **Py***rope* framework for **correct-by-construction** hardware modeling.

- Facilitated **correct-by-construction** hardware modeling purely in Python
- Encoded the correctness proof of (multi-)montgomery reduction algorithm in Python and **verified successfully by compiling to Dafny**
- Unified "sources of truth" for correctness proofs and programming model implementations

UWPLSE, Seattle, WA — Oct. 2019 – Sep. 2021

Research Assistant (PL/Compiler)

Responsible for conducting research with Prof. Zachary Tatlock, specifically,

- Implemented evaluations in the Dynamic Tensor Rematerialization project
- Designed a flexible matching algorithm for domain-specific language compilers.
- Led research projects with other undergraduate students
- Attended and presented at reading groups

Selected Projects & Contributions

CatsTail: Synthesizing Packet Programs via Equality Saturation (**Rust**) GitHub
Music Scores: Reverse engineering of some arrangements (**Lilypond**) GitHub
flexmatch: Flexible offload pattern matching for DNNs (**Python**, **Rust**) GitHub
egg-taichi: Towards automated super-optimization for Taichi programs (**Rust**) GitHub
taichi⋆: High-performance parallel computing in Python (**C++**, **Python**) GitHub
Glenside⋆: Term rewriting for tensor programs (**Rust**) GitHub
veripy: auto-active verification for Python programs (**Python**) GitHub
dtlc: Dependently-typed lambda calculus (**OCaml**) GitHub
Sager: A demonic graph synthesizer for worst-case performance (Rosette, **Racket**) GitHub
Ruxl⋆: Applicatives, Monads and a "*Future*" for Rust (**Rust**) GitHub
Lambda Calculus: UTLC, STLC and System F in OCaml (**OCaml**) GitHub
SimGE: DTR for memory movement optimization (**Rust**) GitHub
Multi-Paxos: Implementation of Multi-Paxos in Java (**Java**) CSE 452
ETH Client: Implementation of an ETH-like Blockchain (**Rust**) COS 471

More on my GitHub

⋆ : Contributor

Teaching

- COS 516: Automated Reasoning about Software (TA, Princeton University)
- CSE 505: Principles of Programming Languages (TA, University of Washington)

Skills

- **Languages:** C/C++, Python, Rust, OCaml, Coq, Dafny, etc. (Open to other languages)
- **Compiler & Applied PL:** Equality Saturation, Static Analysis, Computer-aided Reasoning, SMT
- **PL Theory:** Formal Verification, Type Theory, Mathematical Logic

- **Systems:** Distributed Systems, Machine Learning Systems, Data Center Systems
- **Others:** Algorithms and Data Structures
- **Fun Fact:** I am more seasoned in playing the violin than coding 🎵; I have:
  1. an Lv.9 certificate* (similar to ABRSM Grade 8) issued by Central Conservative of Music;
  2. $>$ **20-year** violin solo experience;
  3. Multiple 1st Prizes (various local competitions in Beijing) and a Silver medal (Beijing regional)†;
  4. 6-year experience with symphony orchestras; 3-year experience as *the Principal 2nd Violinist*;
  5. 3-year experience with a piano quartet/quintet and multiple string quartets (with 1 CD made);
  6. ~ 4 public concerts with a philharmonic orchestra⋆ at the The Giant Egg, Beijing, China.

* : *The highest level for non-professionals*
† : *Awarded during middle and high school years.*
⋆ : *The Beijing National Day School Philharmonic Orchestra*