Mayur Naik

Professor
Department of Computer and Information Science
School of Engineering and Applied Science
University of Pennsylvania

610 Levine Hall | 3330 Walnut St, Philadelphia, PA 19104
Email: mhnaik@cis.upenn.edu
Web: http://www.seas.upenn.edu/~mhnaik/
Phone: +1-215-573-1856

1. Research Interests

My research spans the areas of programming languages and machine learning, with emphasis on general-purpose neurosymbolic approaches that combine deep learning and logical reasoning. My research is motivated by the need to make AI applications more interpretable, data-efficient, safe, and easier to develop.

2. Education

Degree	Year	University	Field
Ph.D.	2008	Stanford University Stanford, CA *Dissertation:* Effective Static Race Detection for Java *Advisor:* Alexander Aiken	Computer Science
M.S.	2003	Purdue University W. Lafayette, IN *Dissertation:* A Type System Equivalent to Model Checking *Advisor:* Jens Palsberg	Computer Science
B.E.	1999	Birla Institute of Technology and Science Pilani, India	Computer Science

3. Appointments

Title	Organization	Years
Professor	University of Pennsylvania	07/2020 – present
Associate Professor	University of Pennsylvania	08/2016 – 06/2020
Assistant Professor	Georgia Institute of Technology	07/2011 – 08/2016
Research Scientist	Intel Research	10/2007 – 05/2011

4. Honors and Awards

4.1. National Awards

- NSF CAREER Award, 2013.
- Microsoft Software Engineering Innovation Foundation (SEIF) Award, 2012.
- Microsoft Research Graduate Fellowship, 2004-2005.

4.2. Publication Awards

- Test-of-Time Paper Award, 2023.
  In 21st ACM SIGSOFT Symposium on the Foundations of Software Engineering (FSE'13) for the paper titled "Dynodroid: An Input Generation System for Android Apps".
- Test-of-Time Paper Award, 2022.
  In 20th ACM SIGSOFT Symposium on the Foundations of Software Engineering (FSE'12) for the paper titled "Automated Concolic Testing of Smartphone Apps".
- Test-of-Time Paper Award, 2021.
  In 6th European Conference on Computer Systems (EuroSys'11) for the paper titled "CloneCloud: Elastic Execution between Mobile Device and Cloud".
- Distinguished Paper Award, 2019.
  In 40th ACM Conference on Programming Language Design and Implementation (PLDI'19) for the paper titled "Continuously Reasoning about Programs using Differential Bayesian Inference".
- Distinguished Paper Award, 2015.
  In 23rd ACM Symposium on Foundations of Software Engineering (FSE'15) for the paper titled "A User-Guided Approach to Program Analysis".
- Distinguished Paper Award, 2014.
  In 35th ACM Conference on Programming Language Design and Implementation (PLDI'14) for the paper titled "On Abstraction Refinement for Program Analyses in Datalog".
- Distinguished Artifact Award, 2013.
  For the paper titled "Dynodroid: An Input Generation System for Android Apps" in Proceedings of the 21st ACM Symposium on Foundations of Software Engineering (FSE'13).
- Distinguished Paper Award, 2009.
  For the paper titled "Effective Static Deadlock Detection" in Proceedings of the 31st International Conference on Software Engineering (ICSE'09).

4.3. University Awards

- Outstanding Junior Faculty Research Award, Georgia Tech, 2016.
  Awarded for "the quality of publications and the significance and impact of research."
- Lockheed-Martin Teaching Excellence Award, Georgia Tech, 2015.
  Awarded for "extraordinary effectiveness in classroom teaching, educational innovations, inspiration transmitted to students, direct impact and involvement with students, and impact on the postgraduate success of students."

5. PUBLICATIONS

5.1. Book Chapters

1. Jens Palsberg and Mayur Naik. ILP-based resource-aware compilation. In *Multiprocessor Systems-on-Chips*. Morgan Kaufmann, 2004.

5.2. Edited Volumes

1. Jennifer Sartor, Mayur Naik, and Chris Rossbach. Proceedings of the 15th ACM SIGPLAN/SIGOPS International Conference on Virtual Execution Environments (VEE 2019), Providence, RI, USA. Co-located with ASPLOS'19, April 2019.

2. Anders Moeller and Mayur Naik. Proceedings of the 4th ACM SIGPLAN International Workshop on the State Of The Art in Java Program Analysis (SOAP 2015), Portland, Oregon, USA. Co-located with PLDI'15, June 2015.

5.3. Journal Articles

1. Elizabeth Dinella, Todd Mytkowicz, Alexey Svyatkovskiy, Christian Bird, Mayur Naik, Shuvendu K. Lahiri. DeepMerge: Learning to Merge Programs. *IEEE Transactions on Software Engineering*, 2022.
2. Haoxian Chen, Chenyuan Wu, Andrew Zhao, Mukund Raghothaman, Mayur Naik, Boon Thau Loo. Synthesizing Formal Network Specifications from Input-Output Examples. *IEEE/ACM Transactions on Networking*, 2022.
3. Yongin Kwon, Byung-Gon Chun, Hayoon Yi, Donghyun Kwon, Seungjun Yang, Sangmin Lee, Ling Huang, Petros Maniatis, Mayur Naik, Yunheung Paek. Mantis: Efficient predictions of execution time, energy usage, memory usage and network usage on smart mobile devices. *IEEE Transactions on Mobile Computing*, 14(10):2059–2072, 2015.
4. David Gay, Joel Galenson, Mayur Naik, Kathy Yelick. Yada: Straightforward parallel programming. *Parallel Computing*, 37(9):592–609, 2011.
5. Mayur Naik and Jens Palsberg. A type system equivalent to a model checker. *ACM Transactions on Programming Languages and Systems*, 30(5), 2008.
6. Mayur Naik and Jens Palsberg. Compiling with code-size constraints. *ACM Transactions on Embedded Computing Systems*, 3(1):163–181, 2004.

5.4. Invited Papers

1. Mayur Naik. Rethinking static analysis by combining discrete and continuous reasoning. In *International Static Analysis Symposium (SAS'19)*, October 2019.
2. Xujie Si, Xin Zhang, Radu Grigore, Mayur Naik. Maximum satisfiability in program analysis: Applications and techniques. In *International Conference on Verification, Model Checking, and Abstract Interpretation (VMCAI'18)*, January 2018.
3. Xujie Si, Xin Zhang, Radu Grigore, Mayur Naik. Maximum satisfiability in software analysis: Applications and techniques. In *International Conference on Computer Aided Verification (CAV'17)*, June 2017.

5.5. Refereed Conference Papers

Note: Author names annotated * indicate co-first authorship.

1. Ziyang Li, Jiani Huang, Jason Liu, Felix Zhu, Eric Zhao, William Dodds, Neelay Velingker, Rajeev Alur, Mayur Naik. Relational Programming with Foundation Models. In *AAAI Conference on Artificial Intelligence (AAAI'24)*, February 2024.
2. Aaditya Naik, Aalok Thakkar, Adam Stein, Rajeev Alur, Mayur Naik. Relational Query Synthesis $\bowtie$ Decision Tree Learning. In *International Conference on Very Large Data Bases (VLDB'24)*, August 2024.
3. Aalok Thakkar, Nathaniel Sands, George Petrou, Rajeev Alur, Mayur Naik, Mukund Raghothaman. Mobius: Synthesizing Relational Queries with Recursive and Invented Predicates. In *ACM Conference on Object-Oriented Programming, Systems, Languages, and Applications (OOPSLA'23)*, October 2023.
4. Hanlin Zhang*, Jiani Huang*, Ziyang Li, Mayur Naik, Eric Xing. Improved Logical Reasoning of Language Models via Differentiable Symbolic Programming. In *Findings of Association for Computational Linguistics (ACL'23)*, July 2023.

5. Aaditya Naik, Yinjun Wu, Mayur Naik, Eric Wong. Do Machine Learning Models Learn Statistical Rules Inferred from Data? In *International Conference on Machine Learning (ICML'23)*, July 2023.
6. Ziyang Li, Jiani Huang, Mayur Naik. Scallop: A Language for Neurosymbolic Programming. In *ACM Conference on Programming Language Design and Implementation (PLDI'23)*, June 2023.
7. Yinjun Wu, Adam Stein, Jacob Gardner, Mayur Naik. Learning to Select Pivotal Samples for Meta Reweighting. In *AAAI Conference on Artificial Intelligence (AAAI'23)*, February 2023. **Oral Presentation**.
8. Elizabeth Dinella, Todd Mytkowicz, Alexey Svyatkovskiy, Christian Bird, Mayur Naik, Shuvendu Lahiri. DeepMerge: Learning to Merge Programs. In *ACM Symposium on Foundations of Software Engineering (FSE'22)*, November 2022.
9. Pardis Pashakhanloo, Aravind Machiry, Hyonyoung Choi, Anthony Canino, Kihong Heo, Insup Lee, Mayur Naik. PacJam: Securing Dependencies Continuously via Package-Oriented Debloating. In *ACM ASIA Conference on Computer and Communications Security (ASIACCS'22)*, May 2022.
10. Pardis Pashakhanloo, Aaditya Naik, Yuepeng Wang, Hanjun Dai, Petros Maniatis, Mayur Naik. CodeTrek: Flexible Modeling of Code using an Extensible Relational Representation. In *International Conference on Learning Representations (ICLR'22)*, April 2022.
11. Jiani Huang*, Ziyang Li*, Binghong Chen, Karan Samel, Mayur Naik, Le Song, Xujie Si. Scallop: From Probabilistic Deductive Databases to Scalable Differentiable Reasoning. In *Conference on Neural Information Processing Systems (NeurIPS'21)*, December 2021.
12. Aaditya Naik, Jonathan Mendelson, Nate Sands, Yuepeng Wang, Mayur Naik, Mukund Raghothaman. Sporq: An Interactive Environment for Exploring Code Using Query-by-Example. In *ACM Symposium on User Interface Software and Technology (UIST'21)*, October 2021.
13. Aalok Thakkar, Aaditya Naik, Nathaniel Sands, Rajeev Alur, Mayur Naik, Mukund Raghothaman. Example-Guided Synthesis of Relational Queries. In *ACM Conference on Programming Language Design and Implementation (PLDI'21)*, June 2021.
14. Ziyang Li, Aravind Machiry, Binghong Chen, Ke Wang, Mayur Naik, Le Song. Arbitrar: User-Guided API Misuse Detection. In *IEEE Symposium on Security and Privacy (S&P'21)*, May 2021.
15. Jonathan Mendelson*, Aaditya Naik*, Mukund Raghothaman, Mayur Naik. GenSynth: Synthesizing Datalog Programs without Language Bias. In *AAAI Conference on Artificial Intelligence (AAAI'21)*, February 2021.
16. Jiani Huang, Calvin Smith, Osbert Bastani, Rishabh Singh, Aws Albarghouthi, Mayur Naik. Generating Programmatic Referring Expressions via Program Synthesis. In *International Conference on Machine Learning (ICML'20)*, July 2020.
17. Xujie Si*, Aaditya Naik*, Hanjun Dai, Mayur Naik, Le Song. Code2Inv: A deep learning framework for program verification. In *Conference on Computer Aided Verification (CAV'20)*, June 2020.
18. Elizabeth Dinella*, Hanjun Dai*, Ziyang Li, Mayur Naik, Le Song, Ke Wang. Hoppity: Learning graph transformations to detect and fix bugs in programs. In *International Conference on Learning Representations (ICLR'20)*, April 2020. **Spotlight Paper**.
19. Mukund Raghothaman, Jonathan Mendelson, David Zhao, Mayur Naik, Bernhard Scholz. Provenance-guided synthesis of Datalog programs. In *ACM Symposium on Principles of Programming Languages (POPL'20)*, January 2020.
20. Xujie Si*, Mukund Raghothaman*, Kihong Heo, Mayur Naik. Synthesizing Datalog programs using numerical relaxation. In *International Joint Conferences on Artificial Intelligence (IJCAI'19)*, August 2019.
21. Kihong Heo*, Mukund Raghothaman*, Xujie Si, Mayur Naik. Continuously reasoning about programs using differential Bayesian inference. In *ACM Conference on Programming Language Design and Implementation (PLDI'19)*, June 2019. **Distinguished Paper Award**.
22. Halley Young, Osbert Bastani, Mayur Naik. Learning neurosymbolic generative models via program synthesis.

In *International Conference on Machine Learning (ICML'19)*, June 2019.

23. Xujie Si*, Yuan Yang*, Hanjun Dai, Mayur Naik, Le Song. Learning a meta-solver for syntax-guided program synthesis. In *International Conference on Learning Representations (ICLR'19)*, May 2019.
24. Xujie Si*, Hanjun Dai*, Mukund Raghothaman, Mayur Naik, Le Song. Learning loop invariants for program verification. In *Conference on Neural Information Processing Systems (NeurIPS'18)*, December 2018. **Spotlight Paper**.
25. Xujie Si*, Woosuk Lee*, Richard Zhang, Aws Albarghouthi, Paris Koutris, Mayur Naik. Syntax-guided synthesis of Datalog programs. In *ACM Symposium on Foundations of Software Engineering (FSE'18)*, November 2018.
26. Kihong Heo*, Woosuk Lee*, Pardis Pashakhanloo, Mayur Naik. Effective program debloating via reinforcement learning. In *ACM Conference on Computer and Communications Security (CCS'18)*, October 2018.
27. Mukund Raghothaman*, Sulekha Kulkarni*, Kihong Heo, Mayur Naik. User-guided program reasoning using Bayesian inference. In *ACM Conference on Programming Language Design and Implementation (PLDI'18)*, June 2018.
28. Woosuk Lee, Kihong Heo, Rajeev Alur, Mayur Naik. Accelerating search-based program synthesis using learned probabilistic models. In *ACM Conference on Programming Language Design and Implementation (PLDI'18)*, June 2018.
29. Xin Zhang, Radu Grigore, Xujie Si, Mayur Naik. Effective interactive resolution of static analysis alarms. In *ACM Conference on Object-Oriented Programming, Systems, Languages, and Applications (OOPSLA'17)*, October 2017.
30. Aws Albarghouthi, Paraschos Koutris, Mayur Naik, Calvin Smith. Constraint-based synthesis of Datalog programs. In *International Conference on Principles and Practice of Constraint Programming (CP'17)*, September 2017.
31. Sulekha Kulkarni, Ravi Mangal, Xin Zhang, Mayur Naik. Accelerating program analyses by cross-program training. In *ACM Conference on Object-Oriented Programming, Systems, Languages, and Applications (OOPSLA'16)*, November 2016.
32. Xujie Si, Xin Zhang, Vasco Manquinho, Mikolas Janota, Alexey Ignatiev, Mayur Naik. On incremental core-guided MaxSAT solving. In *International Conference on Principles and Practice of Constraint Programming (CP'16)*, September 2016.
33. Insu Yun, Changwoo Min, Xujie Si, Yeongjin Jang, Taesoo Kim, Mayur Naik. APISan: Sanitizing API usages through semantic cross-checking. In *USENIX Security Symposium (Security'16)*, August 2016.
34. Ravi Mangal, Xin Zhang, Aditya Kamath, Aditya V. Nori, Mayur Naik. Scaling relational inference using proofs and refutations. In *AAAI Conference on Artificial Intelligence (AAAI'16)*, February 2016.
35. Xin Zhang, Ravi Mangal, Aditya V. Nori, Mayur Naik. Query-guided maximum satisfiability. In *ACM Symposium on Principles of Programming Languages (POPL'16)*, January 2016.
36. Ravi Mangal, Xin Zhang, Aditya V. Nori, Mayur Naik. Volt: A lazy grounding framework for solving very large MaxSAT instances. In *International Conference on Theory and Applications of Satisfiability Testing (SAT'15)*, September 2015.
37. Ghila Castelnuovo, Mayur Naik, Noam Rinetzky, Mooly Sagiv, Hongseok Yang. Modularity in lattices: A case study on the correspondence between top-down and bottom-up analysis. In *International Static Analysis Symposium (SAS'15)*, September 2015.
38. Ravi Mangal, Xin Zhang, Aditya V. Nori, Mayur Naik. A user-guided approach to program analysis. In *ACM Symposium on Foundations of Software Engineering (FSE'15)*, August 2015. **Distinguished Paper Award**.
39. Jongse Park, Hadi Esmaeilzadeh, Xin Zhang, Mayur Naik, William Harris. FLEXJAVA: Language support for safe and modular approximate programming. In *ACM Symposium on Foundations of Software Engineering*

(FSE'15), August 2015.

40. Cong Shi, Karim Habak, Pranesh Pandurangan, Mostafa Ammar, Mayur Naik, Ellen Zegura. COSMOS: Computation offloading as a service for mobile devices. In *ACM Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc'14)*, August 2014.
41. Xin Zhang, Ravi Mangal, Mayur Naik, Radu Grigore, Hongseok Yang. On abstraction refinement for program analyses in Datalog. In *ACM Conference on Programming Language Design and Implementation (PLDI'14)*, June 2014. **Distinguished Paper Award**.
42. Xin Zhang, Ravi Mangal, Mayur Naik, Hongseok Yang. Hybrid top-down and bottom-up interprocedural analysis. In *ACM Conference on Programming Language Design and Implementation (PLDI'14)*, June 2014.
43. Ravi Mangal, Mayur Naik, Hongseok Yang. A correspondence between two approaches to interprocedural analysis in the presence of join. In *European Symposium on Programming (ESOP'14)*, April 2014.
44. Aravind Machiry, Rohan Tahiliani, Mayur Naik. Dynodroid: An input generation system for Android apps. In *ACM Symposium on Foundations of Software Engineering (FSE'13)*, August 2013. **Distinguished Artifact Award**.
45. Yongin Kwon, Sangmin Lee, Hayoon Yi, Donghyun Kwon, Seungjun Yang, Byung-Gon Chun, Ling Huang, Petros Maniatis, Mayur Naik, Yunheung Paek. Mantis: Automatic performance prediction for smartphone applications. In *USENIX Annual Technical Conference (ATC'13)*, June 2013.
46. Xin Zhang, Mayur Naik, Hongseok Yang. Finding optimum abstractions in parametric dataflow analysis. In *ACM Conference on Programming Language Design and Implementation (PLDI'13)*, June 2013.
47. Saswat Anand, Mayur Naik, Hongseok Yang, Mary Jean Harrold. Automated concolic testing of smartphone apps. In *ACM Symposium on Foundations of Software Engineering (FSE'12)*, November 2012.
48. Mayur Naik, Hongseok Yang, Ghila Castelnuovo, Mooly Sagiv. Abstractions from tests. In *ACM Symposium on Principles of Programming Languages (POPL'12)*, January 2012.
49. Percy Liang and Mayur Naik. Scaling abstraction refinement via pruning. In *ACM Conference on Programming Language Design and Implementation (PLDI'11)*, June 2011.
50. Byung-Gon Chun, Sunghwan Ihm, Petros Maniatis, Mayur Naik, Ashwin Patti. CloneCloud: Elastic execution between mobile device and cloud. In *European Conference on Computer Systems (EuroSys'11)*, April 2011.
51. Percy Liang, Omer Tripp, Mayur Naik. Learning minimal abstractions. In *ACM Symposium on Principles of Programming Languages (POPL'11)*, January 2011.
52. Ling Huang, Jinzhu Jia, Bin Yu, Byung-Gon Chun, Petros Maniatis, Mayur Naik. Predicting execution time of computer programs using sparse polynomial regression. In *Conference on Neural Information Processing Systems (NIPS'10)*, December 2010.
53. Pallavi Joshi, Mayur Naik, Koushik Sen, David Gay. An effective dynamic analysis for detecting generalized deadlocks. In *ACM Symposium on Foundations of Software Engineering (FSE'10)*, November 2010.
54. Percy Liang, Omer Tripp, Mayur Naik, Mooly Sagiv. A dynamic evaluation of the precision of static heap abstractions. In *ACM Conference on Object-Oriented Programming, Systems, Languages, and Applications (OOPSLA'10)*, October 2010.
55. Pallavi Joshi, Mayur Naik, Chang-Seo Park, Koushik Sen. Calfuzzer: An extensible active testing framework for concurrent programs. In *Conference on Computer Aided Verification (CAV'09)*, June 2009.
56. Zachary Anderson, David Gay, Mayur Naik. Lightweight annotations for controlling sharing in concurrent data structures. In *ACM Conference on Programming Language Design and Implementation (PLDI'09)*, June 2009.
57. Pallavi Joshi, Chang-Seo Park, Koushik Sen, Mayur Naik. A randomized dynamic program analysis technique for detecting real deadlocks. In *ACM Conference on Programming Language Design and Implementation (PLDI'09)*, June 2009.
58. Mayur Naik, Chang-Seo Park, Koushik Sen, David Gay. Effective static deadlock detection. In *International*

Conference on Software Engineering (ICSE'09), May 2009.

59. Mayur Naik and Alex Aiken. Conditional must not aliasing for static race detection. In *ACM Symposium on Principles of Programming Languages (POPL'07)*, January 2007.
60. Mayur Naik, Alex Aiken, John Whaley. Effective static race detection for Java. In *ACM Conference on Programming Language Design and Implementation (PLDI'06)*, June 2006.
61. Alice X. Zheng, Michael I. Jordan, Ben Liblit, Mayur Naik, Alex Aiken. Statistical debugging: Simultaneous identification of multiple bugs. In *International Conference on Machine Learning (ICML'06)*, June 2006.
62. Ben Liblit, Mayur Naik, Alice X. Zheng, Alex Aiken, Michael I. Jordan. Scalable statistical bug isolation. In *ACM Conference on Programming Language Design and Implementation (PLDI'05)*, June 2005.
63. Mayur Naik and Jens Palsberg. A type system equivalent to a model checker. In *European Symposium on Programming (ESOP'05)*, April 2005.
64. Thomas Ball, Mayur Naik, Sriram K. Rajamani. From symptom to cause: Localizing errors in counterexample traces. In *ACM Symposium on Principles of Programming Languages (POPL'03)*, January 2003.
65. Mayur Naik and Jens Palsberg. Compiling with code-size constraints. In *Conference on Languages, Compilers, and Tools for Embedded Systems (LCTES'02)*, June 2002.

6. PRESENTATIONS

6.1. Invited Talks

- "Learning Interpretable Rules from Structured Data." International Symposium on Practical Aspects of Declarative Languages (PADL'20), January 2020.
- "Rethinking Static Analysis by Combining Discrete and Continuous Reasoning." Static Analysis Symposium (SAS'19), October 2019.
- "Learning to Reason about Programs." Facebook "Probability and Programming" Workshop and Big Code Summit, September 2019.
- "Learning to Reason about Programs." Waterloo ML + Security + Verification Workshop, August 2019.
- "Learning to Reason about Programs." ML4SE Workshop co-located with ICSE'19, May 2019.
- "Learning to Reason about Programs." 2nd Uber Science Symposium, May 2019.
- "Declarative Program Analysis and Big Code: Challenges and Opportunities." Normalized Java Resource Workshop (NJR'18), November 2018.
- "Maximum Satisfiability in Software Analysis: Applications and Techniques." Invited Tutorial, International Conference on Verification, Model Checking, and Abstract Interpretation (VMCAI'18), January 2018.
- "Hunting Software Bugs Using Machine Learning." Distinguished Lecture, Iowa State University, October 2017.
- "Maximum Satisfiability in Software Analysis: Applications and Techniques." Invited Tutorial, International Conference on Computer Aided Verification (CAV'17), July 2017.
- "On Abstraction Refinement for Program Analyses in Datalog." India Software Engineering Conference (ISEC'15), February 2015.
- "Self-Adaptive Static Analysis." 2nd Workshop on Software Correctness and Reliability, ETH Zurich, Switzerland, October 2014.
- "Large-Scale Configurable Static Analysis." 3rd ACM SIGPLAN International Workshop on the State Of The Art in Java Program Analysis (SOAP'14), Edinburgh, UK, June 2014.
- "Automated Testing of Mobile Apps." 1st International Workshop on Challenges in Mobile Apps: A Multi-Disciplinary Perspective, Toronto, Canada, November 2013.
- "Mechanizing Program Analysis with Chord." 1st International Workshop on Learning From Experience (LFX'10),

Toronto, Canada, June 2010.

- "Effective Static Race Detection for Java." Workshop on Architectures and Compilers for Multi-threading, IIT Kanpur, India, December 2007.

6.2. Tutorials

- "Building Program Reasoning Tools using LLVM and Z3." ACM Symposium on Principles of Programming Languages (POPL'20), January 2020.
- "Chord: A Versatile Program Analysis Platform." ACM Conference on Programming Language Design and Implementation (PLDI'11), June 2011.

7. Grants and Contracts

7.1. As Principal Investigator

1. Mayur Naik, Eric Wong, Rajeev Alur. "SHF: Medium: Scallop: A Neurosymbolic Programming Framework for Combining Logic with Deep Learning". NSF CCF #2313010, $1,200,000, Oct 2023 (4 years).
2. Mayur Naik, Rajeev Alur, Mukund Raghothaman. "SHF: Medium: Synthesis of Logic Programs for Democratizing Program Analysis". NSF CCF #2107429, $1,080,000, May 2021 (4 years).
3. Mayur Naik. "Drake: Signature-Guided Detection of Bugs and Vulnerabilities". Air Force Research Laboratory #FA8750-20-2-0501, $170,000, October 2019 (1 year).
4. Mayur Naik. "CODA: A Deep Reinforcement Learning Framework for Building Code Assistants". Facebook Research Award, $50,000, May 2019 (1 year).
5. Mayur Naik. "An AI-based System for Continuously Discovering Software Bugs". Facebook Research Award, $50,000, November 2018 (1 year).
6. Mayur Naik and Le Song. "Syngeries between Program Synthesis and Neural Learning of Graph Structures". NSF CCF #1836936, $900,000, September 2018 (4 years).
7. Mayur Naik, Rajeev Alur, Insup Lee, Oleg Sokolsky, Boon Thau Loo. "ASPIRE: Automatically Subsetting Protocol Implementations Reliably and Efficiently". ONR #N00014-18-1-2021, $6,148,632, January 2018 (5 years).
8. Mayur Naik. "SHF: Small: New Frontiers in Constraint-Based Program Analysis". NSF CCF #1526270, $450,000, September 2015 (3 years).
9. Mayur Naik, Molham Aref, Shan Shan Huang, Jens Palsberg, Tielei Wang. "PetaBlox: Large-Scale Software Analysis and Analytics Using Datalog". DARPA #FA8750-15-2-0009, $3,640,660, October 2014 (5 years).
10. Mayur Naik. "CAREER: Adaptive Large-Scale Program Analysis". NSF CCF #1253867, $484,402, January 2013 (5 years).
11. Mayur Naik. "VIVAS: Verification and Validation of Autonomous Systems". MIT Lincoln Lab, $50,000, November 2016 (1 year).
12. Mayur Naik. "Automated Scalable Testing of Mobile Apps Using Z3". Microsoft Software Engineering Innovation Foundation Award, $25,000, March 2012 (1 year).
13. Mayur Naik. "Enhancing Mobile App Quality via Program Analysis". Google Faculty Research Award, $52,786, December 2011 (1 year).

7.2. As Co-Principal Investigator

1. Shankar Sastry (PI), Sergey Levine, Sanjit Seshia, Claire Tomlin, Somil Bansal, Mayur Naik. "Provably Correct Design of Adaptive Hybrid Neuro-Symbolic Cyber Physical Systems". DARPA #FA8750-23-C-0080,

$5,398,530, June 2023 (4 years).

2. Vincent Liu (PI), Rajeev Alur, Sebastian Angel, Mayur Naik. "FMitF: Track I: Automatic Migration to Serverless Infrastructure, Correctly and Efficiently". NSF CCF # 2124184, $750,000, July 2021 (3 years).
3. Dan Roth (PI), Osbert Bastani, Chris Callison-Burch, Mayur Naik, Irfan Essa, Le Song, Kira Zsolt. "FLASH: Fast Learning via Auxiliary signals, Structured knowledge, and Human expertise". DARPA #FA8750-19-2-0201, $3,349,000, September 2019 (3 years).
4. Taesoo Kim (PI), Emmanouil Antonakakis, Mayur Naik, Wenke Lee. "SaTC-EDU: EAGER: Big Data and Security: Educating The Next-Generation Security Analysts". NSF DGE #1500084, $300,000, June 2015 (3 years).
5. Mostafa Ammar (PI), Irfan Issa, Mayur Naik, Ellen Zegura. "NeTS: Medium: Mobile Computing over Intermittently Connected Networks". NSF CNS #1161879, $695,000, August 2012 (3 years).
6. Hyesoon Kim (PI), Mayur Naik, Santosh Pande. "Smart Offloading Algorithms and Profiling Techniques for Cloud Computing". Samsung, $185,000, May 2012 (1 year).
7. Alex Aiken (PI), Isil Dillig, Thomas Dillig, John Mitchell, Mayur Naik. "STAMP: STatic Analysis of Mobile Programs". DARPA #FA8750-12-2-0020, $3,929,766, January 2012 (4 years).

8. ARTIFACTS

Most of the artifacts resulting from my research are open-source and actively maintained.

8.1. Datasets

1. *DatalogBench*: benchmark suite for interpretable rule learning.
   https://github.com/difflog-project/datalog-bench
2. *ChiselBench*: benchmark suite for C/C++ program debloating.
   https://github.com/aspire-project/chisel-bench

8.2. Software

1. *ProSynth*: tool for synthesizing Datalog programs using constraint solving.
   https://github.com/difflog-project/prosynth
2. *Difflog*: tool for synthesizing Datalog programs using numerical relaxation.
   https://github.com/difflog-project/difflog
3. *Metal*: meta-learning framework for syntax-guided program synthesis.
   https://github.com/petablox/metal
4. *Code2Inv*: deep reinforcement learning framework for inferring loop invariants.
   https://github.com/petablox/code2inv
5. *Chisel*: automated system for debloating C/C++ programs.
   https://github.com/aspire-project/chisel
6. *Euphony*: probabilistic model guided program synthesizer.
   https://github.com/euphony-tool/euphony
7. *APISan*: tool for checking API usage based on semantic anomaly detection.
   https://github.com/sslab-gatech/apisan
8. *Nichrome*: solver for mixed hard and soft constraints.
   https://github.com/nichrome-project/nichrome

9. *SQLtutor*: browser-based tool for teaching and learning SQL.
   https://github.com/sqltutor-project/sqltutor
10. *Dynodroid*: automated input generation system for Android apps.
    https://github.com/dynodroid/dynodroid/
11. *Acteve*: dynamic symbolic execution engine for Android apps.
    https://bitbucket.org/pag-lab/acteve/
12. *Chord*: extensible program analysis framework for Java programs.
    https://bitbucket.org/pag-lab/jchord/
13. *CloneCloud*: system for partitioning and migrating Android apps.
    (Not available publicly; proprietary of Intel Corporation.)
14. *CalFuzzer*: randomized testing tool for concurrent Java programs.
    https://github.com/ksen007/calfuzzer/
15. *CBI*: statistical debugging framework for C programs.
    https://github.com/petablox/cbi

9. TEACHING

9.1. Courses

At University of Pennsylvania:

Semester/Year	*Course Number*	*Course Title*	*Enrollment*	*Effectiveness (0-4)*	
				Instructor	*Course*
Spring 2019	CIS 450/550	Database and Information Systems	124	2.71	2.65
Fall 2018	CIS 700	Software Analysis and Testing	17	3.34	2.94
Spring 2018	CIS 450/550	Database and Information Systems	40	2.96	2.61
Fall 2017	CIS 700	Software Analysis and Testing	11	3.55	3.36
Spring 2017	CIS 450/550	Database and Information Systems	121	1.26	1.44

At Georgia Institute of Technology:

Semester/Year	*Course Number*	*Course Title*	*Enrollment*	*Effectiveness (1-5)*	
				Instructor	*Course*
Spring 2016	CS 6340	Software Analysis and Testing	33	4.4	4.5
Fall 2015	CS 7001	Introduction to Graduate Studies	48	4.4	3.9
Spring 2015	CS 4240	Compilers and Interpreters	39	4.6	4.3
Fall 2014	CS 6340	Software Analysis and Testing	36	4.5	4.2
Spring 2014	CS 4400	Introduction to Database Systems	81	4.0	4.0
Fall 2013	CS 8803	Foundations of Programming Languages	11	4.3	4.3
Spring 2013	CS 4400	Introduction to Database Systems	41	4.2	4.2
Fall 2012	CS 8803	Foundations of Programming Languages	23	4.1	4.1
Fall 2011	CS 6340	Software Analysis and Testing	35	4.0	3.9

9.2. Curriculum Development

In 2015, I developed an online graduate-level course on "Software Analysis". The course covers the theory and practice of software analysis—a body of algorithms and techniques to reason about program behavior with applications to effectively test, debug, and secure large, complex codebases. To date, the course has been taken by over 1,700 students worldwide as part of Georgia Tech's Online Masters in Computer Science degree program.

I am currently revising the course as part of University of Pennsylvania's Online MCIT degree program and plan to offer it regularly starting in Summer 2020. I have redesigned the course to include a series of 12 programming assignments built atop the popular LLVM compiler framework and Z3 constraint solver. The assignments impart hands-on experience with runtime instrumention, fuzzing, dataflow analysis, pointer analysis, constraint generation, type inference, statistical debugging, delta debugging, dynamic symbolic execution, and assertion verification.

10. STUDENTS

10.1. Ph.D. Students

Current

1. **Elizabeth Dinella**, 2018–present
2. **Jiani Huang**, 2018–present
3. **Ziyang Li**, 2019–present
4. **Aaditya Naik**, 2020–present
5. **Adam Stein**, 2021–present
6. **Alaia Solko-Breslin**, 2022–present

Graduated

1. **Xin Zhang**, 2011–2017
   Thesis Topic: Combining Logical and Probabilistic Reasoning in Program Analysis
   Current Position: Assistant Professor, Department of Computer Science and Technology, Peking University
2. **Xujie Si**, 2014–2020
   Thesis Topic: Learning-Aided Program Synthesis and Verification
   Current Position: Assistant Professor, Department of Computer Science, University of Toronto
3. **Sulekha Kulkarni**, 2014–2020
   Thesis Topic: Effective Program Reasoning using Bayesian Inference
   Current Position: Principal Software Engineer, Microsoft
4. **Pardis Pashakhanloo**, 2018–2023
   Thesis Topic: Integrating Declarative Static Analysis with Neural Models of Code
   Current Position: Senior Software Engineer, CertiK
5. **Aalok Thakkar**, 2018–2023
   Co-advised with Rajeev Alur
   Thesis Topic: Example-Guided Synthesis of Relational Queries
   Current Position: Research Scientist, Aptos Labs

10.2. Postdoctoral Scholars

Current

1. **Yinjun Wu**, 2022–present

Graduated

1. **Woosuk Lee**, 2016–2018
   Current Position: Assistant Professor, College of Computing, Hanyang University, Korea

2. **Kihong Heo**, 2017–2019
   Current Position: Assistant Professor, School of Computing, KAIST, Korea
3. **Mukund Raghothaman**, 2017–2019
   Current Position: Assistant Professor, Department of Computer Science, University of Southern California
4. **Aravind Machiry**, 2020–2020
   Current Position: Assistant Professor, Department of Electrical and Computer Engineering, Purdue University
5. **Yuepeng Wang**, 2020–2021
   Co-advised with Rajeev Alur
   Current Position: Assistant Professor, School of Computing Science, Simon Fraser University, Canada

11. SERVICE

11.1. PC Chair/Co-chair

- PC chair, ACM Workshop on Machine Learning and Programming Languages (MAPL), 2020.
- PC co-chair, ACM International Conference on Virtual Execution Environments (VEE), 2019.
- PC co-chair, 4th International Workshop on the State Of The Art in Program Analysis (SOAP), 2015.

11.2. Conference Program Committee

- Senior PC member, International Conference on Computer Aided Verification (CAV), 2019.
- ERC member, ACM Conference on Programming Language Design and Implementation (PLDI), 2019.
- PC member, ACM Symposium on Principles of Programming Languages (POPL), 2019.
- PC member, USENIX Workshop on Hot Topics in Cloud Computing (HotCloud), 2018.
- PC member, International Conference on Computer Aided Verification (CAV), 2018.
- PC member, Workshop on Forming an Ecosystem Around Software Transformation (FEAST), 2017.
- PC member, ACM Conference on Programming Language Design and Implementation (PLDI), 2017.
- PC member, International Static Analysis Symposium (SAS), 2016.
- ERC member, ACM Symposium on Principles of Programming Languages (POPL), 2016.
- PC member, Haifa Verification Conference (HVC), 2015.
- PC member, ACM Conference on Object-Oriented Programming, Systems, Languages, and Applications (OOPSLA), 2015.
- PC member, ACM Symposium on Principles of Programming Languages (POPL), 2015.
- PC member, Haifa Verification Conference (HVC), 2014.
- PC member, India Software Engineering Conference (ISEC), 2014.
- PC member, ACM Workshop on Memory Systems Performance and Correctness (MSPC), 2013.
- PC member, ACM Conference on Programming Language Design and Implementation (PLDI), 2013.
- PC member, ACM Symposium on Principles and Practice of Parallel Programming (PPoPP), 2013.
- ERC member, ACM Symposium on Principles of Programming Languages (POPL), 2012.
- PC member, International Conference on Runtime Verification (RV), 2011.
- PC member, ACM Conference on Object-Oriented Programming, Systems, Languages, and Applications (OOPSLA), 2011.
- PC member, New Ideas and Emerging Results (NIER) track, International Conference on Software Engineering (ICSE), 2010.

- ERC member, ACM Conference on Programming Language Design and Implementation (PLDI), 2010.

11.3. Journal Reviewing

- ACM Transactions on Software Engineering and Methodology, 2017
- ACM Transactions on Computer Systems, 2015
- ACM Transactions on Programming Languages and Systems, 2006, 2008–2010, 2015, 2016
- IEEE Transactions on Cloud Computing, 2015
- IEEE Transactions on Computers, 2009
- IEEE Transactions on Software Engineering, 2006