Malavika Samak

Massachusetts Institute of Technology
G740, 32 Vassar St, Cambridge, MA 02139
malavika@csail.mit.edu
www.malavikasamak.com

Research Interests

I am interested in designing techniques, tools, and workflows to improve developer productivity and software reliability. I have designed program analyses that enable developers to discover, reason, customize, and adapt code to effectively build defect-free software systems.
tags: program analysis, concurrency, test generation, program synthesis, code search

Work Experience

CSAIL, MIT, Cambridge, USA — 2017 – Present
Postdoctoral Associate

Microsoft Research, Bangalore, India — 2017
Visiting Researcher

Google, Mountain View, USA — 2016
Software Engineering Intern

NDS, Bangalore, India — 2010 – 2012
Software Engineer

Education

Indian Institute of Science (IISc), Bangalore, India — 2012 – 2017
Ph.D. in Computer Science
Thesis: Targeted Client Synthesis for Detecting Concurrency Bugs

SJCE, Mysore, India — 2006 – 2010
B.E. in Computer Science and Engineering

Publications

Summary: Published 9 peer-reviewed papers, 7 conference/journal papers and 2 tool/demo papers, in the following venues: POPL, SOSP, PLDI, OOPSLA, FSE, PPoPP and SPLASH.

Conference and Journal Publications

POPL Synthesizing Replacement Classes
Malavika Samak, Deokhwan Kim and Martin C. Rinard
47th ACM SIGPLAN Symposium on Principles of Programming Languages, 2020, Acceptance: 27.5% (68/247)

SOSP Optimizing Big-Data Queries Using Program Synthesis
Matthias Schlaipfer, Kaushik Rajan, Akash Lal and **Malavika Samak**
26th ACM Symposium on Operating Systems Principles, 2017, Acceptance: 16.8% (39/232)

OOPSLA Directed Synthesis of Failing Concurrent Executions
Malavika Samak, Omer Tripp and Murali Krishna Ramanathan
ACM SIGPLAN International Conference on Object-Oriented Programming, Systems, Languages, and Applications, 2016, Acceptance: 25.6% (52/203)

PLDI Synthesizing Racy Tests
Malavika Samak, Murali Krishna Ramanathan and Suresh Jagannathan
ACM SIGPLAN Conference on Programming Language Design and Implementation, 2015, Acceptance: 19.1% (58/303), Artifact Evaluated, Top 10 Video Abstract.

FSE Synthesizing Tests for Detecting Atomicity Violations
Malavika Samak and Murali Krishna Ramanathan
ACM SIGSOFT Symp. on the Foundations of Software Engineering, 2015, Acceptance: 25.4% (74/291), Artifact Evaluated

OOPSLA Multithreaded Test Synthesis for Deadlock Detection
Malavika Samak and Murali Krishna Ramanathan
ACM SIGPLAN International Conference on Object-Oriented Programming, Systems, Languages, and Applications, 2014, Acceptance: 28.6% (53/185), Artifact Evaluated

PPoPP Trace Driven Dynamic Deadlock Detection and Reproduction
Malavika Samak and Murali Krishna Ramanathan
ACM SIGPLAN Symposium on Principles and Practice of Parallel Programming, 2014, Acceptance: 15.6% (28/179)

Tool/Demo Papers

FSE Omen+: A Precise Dynamic Deadlock Detector for Multithreaded Java Libraries
Malavika Samak and Murali Krishna Ramanathan
ACM SIGSOFT Symp. on Foundations of Software Engineering, 2014

SPLASH Omen: A Tool for Synthesizing Tests for Deadlock Detection
Malavika Samak and Murali Krishna Ramanathan
ACM SIGPLAN Conference on Systems, Programming, and Applications: Software for Humanity, 2014

Archives and Technical Reports

Archive Searching for Replacement Classes
Malavika Samak, Jose Pablo Cambronero and Martin C. Rinard
Under Submission, 2021

Technical Report Clearscope: Full Stack Provenance Graph Generation for Transparent Computing on Mobile Devices
Michaell Gordon, Jordan Eikenberry, Anthony Eden, Jeffrey Perkins, **Malavika Samak**, Henny Sipma and Martin C. Rinard
Massachusetts Institute of Technology, Cambridge, United States, 2020

Research Projects

Synthesizing Verified Adapters

Designed and implemented a system, which receives a pair of Java classes as input and automatically synthesizes an adapter class that implements the same interface as the first input class by using the APIs offered by the second input class. The approach leverages the strengths of symbolic execution, constraint solving and program synthesis to construct adapter classes.

Synthesizing Multithreaded Tests

Designed and implemented the first set of algorithms that automatically generate targeted multithreaded tests for detecting concurrency bugs in Java libraries. The automatically synthesized tests constructed by the synthesizers helped expose more than 300 concurrency bugs, including bugs in popular Java libraries.

Code Search for Java Classes

Designed a new technique and implemented a system, ClassFinder, for automatically finding Java classes. Given a query class, ClassFinder automatically searches large code bases to identify and rank potential classes that can act as drop-in replacement to the query class by combining two complementary techniques: embedding-based class ranking and method compatibility matching. Evaluation on ≈600 thousand open-source classes demonstrates that ClassFinder can effectively find appropriate classes.

Talks

Invited talks	**Rising Stars in Computer Science Series**, U. Mass. Amherst, 2019 **Microsoft Research**, Bangalore, 2017 **Google**, Mountain View, 2016 **Dagstuhl Seminar** on Concurrency (Tutorial), 2016 **DRDO-IISc** workshop on verification of System Software, IISc, 2016 **Hewlett Packard Enterprise**, Bangalore, 2015 **Undergraduate Summer School**, IISc, 2014
Conference talks	**POPL**: Synthesizing Replacement Classes, 2020 **OOPSLA**: Directed Synthesis of Failing Concurrent Executions, 2016 **FSE**: Synthesizing Tests for Detecting Atomicity Violations, 2015 **PLDI**: Synthesizing Racy Tests, 2015 **OOPSLA**: Multithreaded Test Synthesis for Deadlock Detection, 2014 **PPoPP**: Trace Driven Deadlock Detection and Reproduction, 2014

Awards and Honors

- Invited speaker for **Rising Stars in Computer Science Seminar**, University of Massachusetts, Amherst, 2019.
- Invited to attend the 4^{th} **Heidelberg Laureate** Forum, 2016.
- Invited to the **Dagstuhl Seminar** on concurrency, 2016.
- Received **Google PhD fellowship**, 2015.

- PLDI video abstract voted in the Top 10 by the attendees, 2015.
- Invited to speak at Google Test Automation Conference, 2015
- Secured All India Rank 107 (out of 156,780 candidates) in Graduate Aptitude Test Entrance (GATE), 2012.
- President Award for Girl Scouts, Government of India, 2004.

Teaching, Grants and Mentorship

- Co-organized the Programming Languages Mentoring Workshop (PLMW) at SPLASH 2020 with Karim Ali and Jonathan Bell. The event received over 120 student applications, that included 40 women applicants.
- Presented in AMP-DARPA engagement and contributed to HACCS and SafeDocs DARPA grant proposals.
- Teaching assistant for the graduate level course on Operating Systems, IISc.
- Guest lecturer for the graduate level course on Software Engineering, IISc.
- Mentored four undergraduate summer interns between 2014-16. They subsequently joined graduate programs at Carnegie Mellon University and Stanford University.
- Co-organized Technologix 2009, a three day national level computer science symposium conducted by Computer Society of India, SJCE.

Service

Co-chair	**PLMW:** Programming Languages Mentoring Workshop, SPLASH 2020 **AEC:** Artifact Evaluation Committee, PPoPP 2018
Program Committee	**PLDI**: Programming Languages, Design and Implementation, 2022 **SC**: Super Computing, 2021 **ICCQ**: International Conference on Code Quality, 2021 **ICPP**: International Conference on Parallel Processing, 2020 **SC**: Super Computing, 2019 **PPoPP**: Principles and Practices of Parallel Programming, 2019 **OOPSLA**: Object Oriented Programming, Systems, Languages, and Applications, 2018
Journal Reviewer	**TOPLAS**: Transactions on Programming Languages and Systems, 2021 **TSE**: Transactions on Software Engineering, 2019
External Program Committee	**OOPSLA**: Object Oriented Programming, Systems, Languages, and Applications, 2019 **PPoPP**: Principles and Practices of Parallel Programming, 2018
Reviewer	**SRC**, Student Research Competition, SPLASH 2021 **SRC**, Student Research Competition, PLDI 2018

Artifact Evaluation Committee

PLDI: Programming Languages Design and Implementation, 2017
PPoPP: Principles and Practices of Parallel Programming, 2017
PLDI: Programming Languages Design and Implementation, 2016
OOPSLA: Object Oriented Programming, Systems, Languages, and Applications, 2016
PPoPP: Principles and Practices of Parallel Programming, 2016
POPL: Principles of Programming Languages, 2016
OOPSLA: Object Oriented Programming, Systems, Languages, and Applications, 2015