Akshay Anand Utture

Joining Uber's Programming Systems Group in July 2023 as Software Engineer II
Research areas of interest: Static Analysis, Compilers and Verification
Ph.D. advisor: Jens Palsberg
Email: akshay.utture@gmail.com

EDUCATION

University of California, Los Angeles *(Sept 2018 - June 2023)*

PhD. in Computer Science (GPA: 3.788/4.0)

Indian Institute of Technology Madras, Chennai, India *(July 2013 - May 2018)*

5 year Integrated Btech. & Mtech. in Computer Science and Engineering (GPA: 9.16/10.0)

INDUSTRIAL INTERNSHIPS

Amazon Web Services - Boston, USA - *Applied Scientist Intern (June 2022 - Sept 2022)*

Worked with the Automated Reasoning Group on the CBMC (C bounded model checker) tool and proved unbounded memory safety for the FreeRTOS task scheduler.

Google - Sunnyvale, USA (remote) - *Ph.D. Software Eng. Intern (June 2020 - Sept 2020)*

Worked on an Android Static Analysis tool to improve the precision of its call-graph analysis. The solution involved designing a pass to selectively apply context sensitivity.

RESEARCH EXPERIENCE

Automated Repair for Resource Leak Warnings *(Jan 2022 - June 2023)*

Advisor: Jens Palsberg, UCLA

- Designed a specialized repair tool that generates high-quality fixes for resource-leaks. The tool introduces a new static analysis to separate out leaks that are infeasible to fix, and for the remaining leaks, constructs fixes that preserve the program semantics. Submitted to FSE'23.

Application Code Analysis using a Partial Library *(Jan 2021 - Dec 2021)*

Advisor: Jens Palsberg, UCLA

- Designed a pre-processor for static analyses that speeds up the analysis of application code. The pre-processor uses a program-slicing based analysis to pick a small part of the library that is most relevant to the analysis queries in the application code.
- Accepted at ICSE 2022 [1].

Pruning False Positives Edges from a Static Call-graph *(Jan 2019 - Dec 2021)*

Advisor: Jens Palsberg, UCLA

- Designed a tool that learns to prune false-positive static call-graph edges by applying machine-learning to a large repository of Java static and dynamic call graphs. Also, showed that this pruned call-graph reduces false-positives in two downstream client analyses.
- Accepted at ICSE 2022 [2].

Efficient lock-step synchronization in task-parallel languages *(April 2017 - July 2018)*

Advisor: Krishna Nandivada, IIT Madras

- Designed a new scheduling mechanism for the Clocks construct in the X10 task-parallel programming language, and proved why the new mechanism was better.
- Published in the 'Software Practice and Experience' journal [3].

TECHNICAL SKILLS

Analysis and verification tools: Wala (most experience), JavaParser, CBMC.

Compilers Coursework: Compiler Design (IITM 2015), Program Analysis (IITM 2017), Static Analysis (UCLA 2020), Types and Programming Languages (UCLA 2020)

Programming Languages: Java, Python, C++, C, Javascript, Haskell

ML Libraries: Scikit-learn, Keras

CS Background: Compilers, Architecture, Operating Systems, Security, Networks, Databases

Math Background: Graph Theory, Logic, Linear Algebra, Probability and Statistics

TEACHING EXPERIENCE

Compiler Construction (Teaching Assistant) - *UCLA (Fall '19, Fall '20, Fall '21)*

PUBLICATIONS

[1] Fast and Precise Application Code Analysis using a Partial Library

Akshay Utture and Jens Palsberg. ICSE'22 (https://doi.org/10.1145/3510003.3510046).

[2] Striking a Balance: Pruning False-Positives from Static Call Graphs

Akshay Utture, Shuyang Liu, Christian Kalhauge and Jens Palsberg. ICSE'22 (https://doi.org/10.1145/3510003.3510166).

[3] Efficient lock-step synchronization in task-parallel languages

Akshay Utture and V Krishna Nandivada. Software: Practice and Experience 49 (9), 1379-1401, 01 July 2019 (https://doi.org/10.1002/spe.2726).

[4] Student Course Allocation with Constraints

Akshay Utture, Vedant Somani, Prem Krishnaa, Meghana Nasre. SEA 2019. Lecture Notes in Computer Science, vol 11544. Springer, Cham (https://doi.org/10.1007/978-3-030-34029-2_4).