DANESHVAR AMROLLAHI

daneshvar@cs.stanford.edu cs.stanford.edu/~daneshvar github.com/daneshvar-amrollahi
353 Jane Stanford Way, Gates Computer Science #481, Stanford, CA 94305

EDUCATION

- **Stanford University** *2024/01 - Present*
  PhD in Computer Science
- **University of Tehran** *2018/09 - 2023/02*
  BSc in Computer Engineering (Software) *GPA: 18.02/20.00*

RESEARCH INTERESTS

- Automated Reasoning
- Verification
- Satisfiability Modulo Theories (SMT)
- Computer Systems

PUBLICATIONS

- D. Amrollahi, E. Bartocci, G. Kenison, L. Kovács, M. Moosbrugger, M. Stankovič (2022). **Solving Invariant Generation for Unsolvable Loops**. *29th International Static Analysis Symposium.* **Awarded the Radhia Cousot Young Researcher Best Paper Award.**
- A. Humenberger, D. Amrollahi, N. Bjørner, L. Kovács (2022). **Algebra-Based Reasoning for Loop Synthesis**. *Formal Aspects of Computing.*
- D. Amrollahi, H. Hojjat, P. Rümmer (2023). **An Encoding for CLP Problems in SMT-LIB**. *10th Workshop on Horn Clauses for Verification and Synthesis.*
- D. Amrollahi, E. Bartocci, G. Kenison, L. Kovács, M. Moosbrugger, M. Stankovič (2023). **(Un)Solvable Loop Analysis**. *Submitted to Formal Methods in System Design.*
- P. Hozzová, D. Amrollahi, M. Hajdu, L. Kovács, A. Voronkov, E.M. Wagner (2024). **Synthesis of Recursive Programs in Saturation.** *Submitted.*

RESEARCH EXPERIENCE

- **Center for Automated Reasoning, Stanford University** *Stanford, United States*
  Under Prof. Clark Barrett *2024/01 - Present*
  Enhancing the performance robustness of cvc5, an SMT solver, against benchmark scrambling (assertion shuffling, symbol renaming, etc) by adding a new pre-processing pass.
- **Research Intern at Automated Program Reasoning Group, TU Wien** *Vienna, Austria*
  Under Prof. Laura Kovács and Prof. Ezio Bartocci *2021/07 - 2022/02 + 2023/05 - 2023/12*
  Worked on different topics including polynomial loop invariant generation, program synthesis, symbolic computation, probabilistic programming, saturation-based theorem proving, structural induction, etc.
- **Research Intern at Dependable Systems Lab, EPFL** *Lausanne, Switzerland*
  Under Prof. George Candea *2022/07 - 2022/08*
  Integrated Z3's support for quantifiers in first-order logic into KLEE's source code, to mitigate the path explosion issue in symbolic execution due to loops (e.g., libc strings functions), by using loop summaries.
- **Research Intern at Programming Methodology Group, ETH Zürich** *Zürich, Switzerland*
  Under Prof. Peter Müller *2022/03 - 2022/04*
  Worked on devising a methodology for verification and specification of Golang programs that use global variables and package initialization code, using separation logic.

TEACHING EXPERIENCE

- **Teaching Assistant**
  Department of Electrical and Computer Engineering, University of Tehran
  - **Advanced Programming**. *Fall 2020, Spring 2021, Fall 2021*
  - **Data Structures**. *Fall 2020*
  - **Design and Analysis of Algorithms**. *Spring 2021*
  - **Discrete Mathematics**. *Spring 2020, Fall 2020, Spring 2021*
  - **Engineering Probability and Statistics**. *Spring 2021*
  - **Operating Systems**. *Spring 2022, Fall 2022*

HONORS AND AWARDS

- **Stanford School of Engineering (SoE) Fellowship** *2024*
- **Radhia Cousot Young Researcher Best Paper Award** *2022/12*
  29th Static Analysis Symposium (SAS 2022). *Auckland, New Zealand*
- **Ranked 8th** in Regional Contest of **ACM-ICPC** West Asia Region, Tehran site. *2020*

PROJECTS

- **Vampire** — github.com/vprover/vampire/tree/synthesis-recursive *C++*
  Implemented a framework within a saturation-based first-order theorem prover for synthesizing recursive programs using structural induction over algebraic datatypes, and superposition calculus.
- **Polar** — github.com/probing-lab/polar *Python, SymPy*
  Implemented a polynomial loop-invariant synthesizer for (probabilistic) unsolvable loops, using recurrences.
- **KLEE with quantifiers** — github.com/bolt-perf-contracts/klee/pull/9 *C++, Z3*
  Integrated Z3's support for existential/universal quantifiers into the KLEE symbolic execution engine codebase to summarize loops using quantified formulas in first-order logic, and mitigate the path explosion problem.
- **Koloocheh** — github.com/daneshvar-amrollahi/Koloocheh *Python, gRPC*
  A peer-to-peer file-sharing system employs the flooding algorithm for search operations and ensures a small graph diameter by leveraging random graph properties.

SKILLS

- **Programming Languages**:
  - Experienced in C, C++, Python.
  - Familiar with Scala, Go, Bash.
- **Tools**: cvc5, Z3, KLEE, LaTeX.