Fabian Ritter

Campus E1 3, Rm. 403, 66123 Saarbrücken, Germany
fabian.ritter@cs.uni-saarland.de

Education

since 2017 | **PhD student** in the International Max Planck Research School for Computer Science, at the Compiler Design Lab at Saarland University

2016 – 2017 | **Doctoral Preparatory Phase** at Saarbrücken Graduate School of Computer Science, Saarland University

2012 – 2015 | **Bachelor of Science** in Computer Science with minor in Mathematics, Saarland University

2004 – 2012 | **Abitur** at Leibniz Gymnasium, St. Ingbert

Scholarships and Certificates

since 2017 | Fellow of the International Max Planck Research School for Computer Science

2016 – 2017 | Scholarship holder of Saarbrücken Graduate School of Computer Science, Saarland University

2013 – 2015 | Member of Bachelor Förderprogramm, Saarland University

Academic Activities

2017 – 2019 | Lecturer's assistant at Saarland University for the lecture Compiler Construction

2014 – 2017 | Tutor/Teaching assistant at Saarland University for the lectures:
- ▷ Compiler Construction
- ▷ Nebenläufige Programmierung (concurrent programming)
- ▷ Grundzüge der Theoretischen Informatik (theoretical computer science)
- ▷ Systemarchitektur (system architecture)

Skills

Language
- ▷ German: native
- ▷ English: fluent

Technical
- ▷ C/C++, Python: experienced, used in research projects, course work, private projects, especially using the LLVM compiler framework
- ▷ Scala: used in research projects and private projects
- ▷ Java, C#, Verilog: used in course work
- ▷ LaTeX: used for writing academic documents, designing slides and posters

Research Projects

2018– **Inferring Port Mappings of Out-of-Order Processors** (current research)

- ▷ Inferring the instruction-to-execution-port mapping of modern out-of-order processors by Intel, AMD, and ARM from experiments with time measurements
- ▷ Exploring mechanisms for experiment design and mapping inference using formal methods as well as learning-based approaches

2017– **Memory Safety in C** (current research)

- ▷ Understanding memory-safety-induced vulnerabilities as a problem of the programming language definition (rather than an eternal sequence of attacks and counter-measures)
- ▷ Exploring ways of making C a safe programming language

2016 **Supporting Transcendental Functions in Daisy, a Sound Verification Tool for the Precision of Floating-Point Computations**
(Research Immersion Lab, Automated Verification and Approximation group, Max Planck Institute for Software Systems)

- ▷ Extended Daisy for soundly estimating round-off errors caused by using floating-point operations for trigonometric and exponential functions
- ▷ Developed algorithms for obtaining sound rational bounds for real-valued results of transcendental functions

2016 **Memory Safety Analysis in Sprattus**
(Research Immersion Lab, Real-Time and Embedded Systems Lab, Saarland University)

- ▷ Implemented analysis domains for obtaining information about accessed memory ranges and allocated memory regions in our symbolic abstraction framework
- ▷ Designed a structured memory model for LLVM bitcode for use in symbolic abstraction
- ▷ Evaluated on benchmarks from the Software Verification Competition 2016 with promising results

2015 **Compiler Optimizations using Symbolic Abstraction**
(Bachelor's Thesis, Compiler Design Lab, Saarland University)

- ▷ Extended a framework for static analysis of LLVM bitcode by symbolic abstraction
- ▷ Implemented classical compiler transformations in the clang compiler based on the found analysis results
- ▷ Investigated how combining these analyses influences transformation quality