CFV'15: Ninth International Workshop on Constraints in Formal Verification, Thursday, November 5, 2015

Program Schedule

General Chair: Miroslav Velev (Aries Design Automation, U.S.A.)

Workshop Chair: Alex Groce (Oregon State University, U.S.A.)

Publicity Chair: Kristin Yvonne Rozier (University of Cincinnati, U.S.A.)

Program Committee: Masahiro Fujita (University of Tokyo, Japan)
Arnaud Gotlieb (Simula Research Laboratory, Norway)
Jie-Hong R. Jiang (National Taiwan University, Taiwan)
Kristin Yvonne Rozier (University of Cincinnati, U.S.A.)
Andreas Veneris (University of Toronto, Canada)
Robert Wille (University of Bremen & DFKI GmbH, Germany)

9:00 – 10:30 Session 1

Session Chair: Miroslav Velev (Aries Design Automation, U.S.A.)

9:00 – 10:00 Invited Talk: On the Probabilistic Analysis of Software at NASA Ames

Corina Pasareanu (CMU/NASA Ames, U.S.A.)

Abstract: Probabilistic software analysis aims at quantifying how likely a target event is to occur, given a probabilistic characterization of the behavior of a program or of its execution environment. Examples of target events may include an uncaught exception, the invocation of a certain method, or the access to confidential information.We are working on a symbolic execution approach to probabilistic software analysis that first computes the conditions to reach the target event, and then tries to quantify the fraction of the input domain satisfying these conditions. Unlike past approaches, that were mostly performed at model level, and were thus only applicable to early software design stages or required explicit (and hard to maintain) abstraction from the code, our techniques are performed directly at the code level. Our techniques are built on top of the Symbolic PathFinder symbolic execution tool. We discuss applications to the analysis of Unmanned Aerial Systems developed at NASA Ames.

Short Bio: Corina Pasareanu is a researcher in software engineering at NASA Ames, in the Robust Software Engineering group. She is employed by Carnegie Mellon University, at the Silicon Valley campus. She is best known for her work on symbolic execution and the application of automated learning techniques to assume-guarantee reasoning. Most recently, she has worked on scalable probabilistic analysis of software, through compositional, abstraction and symbolic techniques. More information about her work can be found at: http://ti.arc.nasa.gov/profile/pcorina

10:00 – 10:30 **Diagnosing Unreachable States Using Property-Directed Reachability**

Ryan Berryhill (University of Toronto, Canada)

Andreas Veneris (University of Toronto, Canada)

Abstract: In the modern design cycle, substantial manual effort is required to correct errors found when verification reveals an unreachable state. This work introduces two methodologies to automate this task. Given an unreachable target state, both methodologies return a set of design locations where changes can be implemented to make the target state reachable (i.e., solutions). The first methodology uses intertwined steps of reachable state-space approximation, property checking, and traditional debugging to compute a subset of the solutions that make the target state reachable in some fixed number of clock cycles. The second methodology uses property-directed reachability with an enhanced version of the circuit's transition relation to compute the complete set of solutions to the problem. As an additional benefit, it returns an inductive invariant proving that no further solution(s) exist. The completeness of the approach comes at the cost of increased runtime when compared to the first methodology. Empirical results on industrial designs confirm the effectiveness of both approaches.

10:30 – 10:40 Coffee Break

10:40 – 12:40 Session 2

Session Chair: Hari Mony (IBM, U.S.A.)

10:40 – 11:10 **Quantified Bounded Model Checking for Rectangular Hybrid Automata**

Luan Viet Nguyen (University of Texas at Arlington, U.S.A.)

Djordje Maksimovic (University of Toronto, Canada)

Taylor T Johnson (University of Texas at Arlington, U.S.A.)

Andreas Veneris (University of Toronto, Canada)

Abstract: Satisfiability Modulo Theories (SMT) solvers have been successfully applied to solve many problems in formal veri-fication such as bounded model checking (BMC) for many classes of systems from integrated circuits to cyber-physical systems (CPS). Typically, BMC is performed by checking satisfiability of a possibly long, but quantifier-free formula. However, BMC problems can naturally be encoded as quantified formulas over the number of BMC steps. In this approach, we then use decision procedures supporting quantifiers to check satisfiability of these quantified formulas. This approach has previously been applied to perform BMC using a Quantified Boolean Formula (QBF) encoding for purely discrete systems, and then discharges the QBF checks using QBF solvers. In this paper, we present a new quantified encoding of BMC for rectangular hybrid automata (RHA), which requires using more general logics due to the real (dense) time and real-valued state variables modeling continuous states. We have implemented a preliminary experimental proto-type of the method using the HyST model transformation tool to generate the quantified BMC (QBMC) queries for the Z3 SMT solver. We describe experimental results on several timed and hybrid automata benchmarks, such as the Fischer and Lynch-Shavit mutual exclusion algorithms. We compare our approach to quantifier-free BMC approaches, such as those in the dReach tool that uses the dReal SMT solver, and the HyComp tool built on top of nuXmv that uses the MathSAT SMT solver. Based on our promising experimental results, QBMC may in the future be an effective analysis approach for RHA as further improvements are made in quantifier handling in SMT solvers such as Z3.

11:10 – 11:40 **Enhancing Symbolic Execution for Coverage-Oriented Testing**

Sébastien Bardin (CEA LIST, France)

Nikolay Kosmatov (CEA LIST, France)

Mickaël Delahaye (CEA LIST, France)

Abstract: Automatic (code-based) test data generation is a major topic in software engineering, and Dynamic Symbolic Execution (DSE) is a very promising approach to this problem. However, while DSE inherently covers feasible paths of the programs under test, practical testing is more concerned with fulfilling so called “coverage criteria,” such as instruction coverage, branch coverage, MCDC (in aeronautic) or mutations. Two issues arise for DSE in this context: first, path coverage may be not adapted to the criteria under consideration, second, some of the coverage requirements may be infeasible. We propose two ways of taming these problems. First, we define a variant of DSE which handles explicit coverage requirements at only a reasonable cost (i.e., a polynomial growth of the search space, while a standard approach results in an exponential blowup). Second, we use a combination of static analyses (abstract interpretation and weakest precondition) in order to detect infeasible coverage requirements. These results have been implemented into the LTest plugin of the open-source

Frama-C software analyzer. We also present new experimental results on weak forms of the MCDC criterion, and additional optimizations of the approach.

11:40 – 12:10 String Constraint Solving with Logic Circuit Manipulation

Hung-En Wang (National Taiwan University, Taiwan)

Tzung-Lin Tsai (National Taiwan University, Taiwan)

Chun-Han Lin (National Chengchi University, Taiwan)

Fang Yu (National Chengchi University, Taiwan)

Jie-Hong R. Jiang (National Taiwan University, Taiwan)

Abstract: Severe security vulnerabilities in web applications are typically due to string manipulation errors that can be resolved via solving string constraints. We present an effective string constraint solver that combines automata-based string analysis with circuit implementation. We evaluate our approach on open source web applications and attack patterns, demonstrating its unique benefits compared to the existing automata-based and SMT-based string solvers.

12:10 – 12:40 Verification and Debugging of UML/OCL Models

Robert Wille (University of Bremen & DFKI GmbH, Germany)

Abstract: Motivated by the ever increasing complexity of today's systems, designers more and more start to use modeling languages such as UML/OCL in order to (formally) describe a system to be realized. Once such a formal specification has been derived, the structure, the behavior, and the properties of the desired system can automatically be checked for correctness -- even in the absence of a precise implementation. Besides structural aspects (Is it indeed possible to instantiate a system considering all constraints?), also precise behavior (e.g. the reachability of bad states, good states, etc.) can be proven. To this end, automatic methods and approaches are available which will be reviewed in this talk. Besides that, also debugging methods are covered that, in case errors in the UML/OCL specification have been detected, help to identify the source of them.

12:40 – 13:30 Lunch Break

(lunch will be provided)

13:30 – 15:00 Session 3

Session Chair: Flavio M. de Paula (IBM, U.S.A.)

13:30 – 14:30 Invited Talk: Lightweight Formal Methods for LLVM Verification

Santosh Nagarakatte (Rutgers University, U.S.A.)

Abstract: Compilers form an integral component of the software development ecosystem. Compiler writers must understand the specification of source and target languages to design sophisticated algorithms that transform programs while preserving semantics. Unfortunately, compiler bugs in mainstream compilers are common. Compiler bugs can manifest as crashes during compilation, or, much worse, result in the silent generation of incorrect programs. Such miscompilations can introduce subtle errors that are difficult to diagnose and generally puzzling to software developers.

The talk will describe the problems in developing peephole optimizations that perform local rewriting to improve the efficiency of LLVM code. These optimizations are individually difficult to get right, particularly in the presence of undefined behavior; taken together they represent a persistent source of bugs. The talk will present Alive, a domain-specific language for writing optimizations and for automatically either proving them correct or else generating counterexamples. A transformation in Alive is shown to be correct automatically by encoding the transformation into constraints, which are automatically checked for validity using a Satisfiability Modulo Theory(SMT) solver. Furthermore, Alive can be automatically translated into C++ code that is suitable for inclusion in an LLVM optimization pass.

Alive is based on an attempt to balance usability and formal methods; for example, it captures—but largely hides—the detailed semantics of three different kinds of undefined behavior in LLVM. We have translated more than 300 LLVM optimizations into Alive and, in the process, found that eight of them were wrong. I will conclude the talk highlighting the lessons learned and the challenges in incorporating lightweight formal methods in the tool-chain of the compiler developer.

Short Bio: Santosh Nagarakatte is an Assistant Professor of Computer Science at Rutgers University. He obtained his Ph.D. from the University of Pennsylvania. His research interests are in Hardware-Software Interfaces spanning Programming Languages, Compilers, and Computer Architecture. His papers have been selected as IEEE MICRO TOP Picks papers of computer architecture conferences in 2010 and 2013. He has received the NSF CAREER Award in 2015, PLDI 2015 Distinguished Paper Award, and the Google Faculty Research Award in 2014 for his research on incorporating lightweight formal methods for LLVM compiler verification.

14:30 – 15:00 **Model-Based Verification of Simulink Generated PLC Programs Using Model Checkers**

Nannan He (Minnesota State University at Mankato Mankato, U.S.A.)

Allen Gale (Minnesota State University at Mankato Mankato, U.S.A.)

Abstract: Programmable Logic Controllers (PLCs) have been widely applied in safety-critical industrial processes. Automated testing of PLC programs is a challenging task for control system engineers. A method of mutation-based testing of Simulink models for verifying PLCs programmed in the Structured Text (ST) language is proposed. In this work, ST programs are assumed to be automatically generated from Simulink models using the latest tool Simulink PLC Coder from Mathworks. We utilize Simulink diagrams as system design models. Simulink is a powerful design tool for developing complex event-driven applications. But, it has the limited capability of verification. Our approach first injects logic and timing related mutations to the Simulink diagram to obtain a set of mutants. Then, a miter model built with original Simulink diagram and its mutant is transformed to a timed automaton network, which is the input format for the Uppaal model checker. To avoid the state space explosion, model slicing is applied to abstract away irrelevant Simulink blocks before transformation. Finally, it uses Uppaal to generate test cases to detect mutants so as to verify the consistency between PLC implementation with the design. The approach is experimentally assessed on a water control system case study.

15:00 – 15:10 Coffee Break

15:10 – 17:10 Session 4

Session Chair: Clayton McDonald (Synopsys, U.S.A.)

15:10 – 15:40 **A CP-Based System for Checking and Generating Memory Consistency Tests**

Vincent Abikhattar (ARM, France)

Laurent Arditi (ARM, France)

Olivier Ponsini (ARM, France)

Suzanne Shoaraee (ARM, France)

Abstract: Nowadays, CPU memory systems become more and more complex. A directed test approach is needed to uncover subtle bugs that random testing misses. Such a directed approach may now be efficiently developed on top of recent works on formal memory models. In this paper, we propose a constraint programming-based implementation of a checker of

litmus tests using an axiomatic formulation of the ARM memory model. From this, we derive a test generator that is embedded into a validation environment of an ARM CPU. These tests target more precisely the memory system than the randomly generated ones, and so they may find violations of the memory model in designs.

15:40 – 16:10 SAT-Based Explicit LTL Reasoning

Jianwen Li (East China Normal University, China & Rice Univ., U.S.A.)

Shufang Zhu (East China Normal University, China)

Geguang Pu (East China Normal University, China)

Moshe Y. Vardi (Rice University, U.S.A.)

Abstract: We present here a new explicit reasoning framework for linear temporal logic (LTL), which is built on top of propositional satisfiability (SAT) solving. As a proof-of-concept of this framework, we describe a new LTL satisfiability algorithm. We implemented the algorithm in a tool, LTLCheck, which is built on top of the Minisat SAT solver. We tested the effectiveness of this approach by demonstrating that LTLCheck significantly outperforms all existing LTL satisfiability solvers. Furthermore, we show that the framework can be extended from propositional LTL to assertional LTL (where we allow theory atoms), by replacing Minisat with the Z3 SMT solver, and demonstrating that this can yield an exponential improvement in performance.

16:10 – 16:40 SMT-Driven Intelligent Storage for Big Data

Eric W. D. Rozier (University of Cincinnati, U.S.A.)

Kristin Y. Rozier (University of Cincinnati, U.S.A.)

Abstract: Big Data presents itself as a double-edged sword: both promising and problematic. While large-scale data collection and analytics are revolutionizing scientific discovery, they also pose serious challenges for infrastructure in cost-constrained environments. NASA's Center for Climate Simulation named data infrastructure the biggest problem facing climate science, with the growth of storage needs outstripping the growth of necessary computing power by 6.67x. New advances in intelligent software-defined data centers (SDDCs) have the potential to combat these challenges, but they must incorporate the ability to optimize data dependence constraints quickly and performably. We formally define data dependence constraints for Big Data storage architectures, present automated methods for encoding the system states of such architectures into constraints suitable for SMT solving with Z3, and describe methods for efficiently resolving our data dependence constraints to enable SDDCs. We include proofs of correctness of our new constructions and optimizations inspired by experimental evaluation.

16:40 – 17:10 Addressing Information Flow Properties for HW Security Verification

Wen Chen (Freescale Semiconductor, U.S.A.)

Monica Farkash (Freescale Semiconductor, U.S.A.)

Jing Huang (Freescale Semiconductor, U.S.A.)

Jay Bhadra (Freescale Semiconductor, U.S.A.)

Abstract: The emergence of Internet-of-Things has imposed enormous challenges on designing and verifying hardware satisfying security requirements. Common security properties such as confidentiality and integrity can be formulated as information flow properties. However, it has been claimed that such properties are difficult to express in popular property languages such as SVA, which hinders its application in hardware verification. This paper studies why such an expression is difficult by providing a formulation of information flow property in CTL and by doing so, revealing the practical difficulty for proving such a property with off-the-shelf formal engines. Moreover, we propose an alternative approach by formulating it as a sequential equivalence checking problem that can be solved with readily available formal engines.