netidee

STIPENDIEN

Automated Software Verification with First-Order Theorem Provers

Progress Report | Call 16 | Stipendium ID 5761

Inhalt

1. Introduction

Automated software verification is a realm mainly depending on satifiability modulo theories (SMT)-solvers. In this dissertation, I'm aiming to make first-order theorem proving more applicable to automated program reasoning and verification by extending program semantics in trace logic and our trace logic based tool - the Rapid Verification Framework. The integral part of the work is automating inductive reasoning which is knowingly hard to do.
Our recent efforts have been relying on manually identified lemmas - coined *trace lemmas* - that are applicable to a wide range of programs containing loops and unbounded data structures as arrays. However, these are limited to a certain type of reasoning and do not always help to find a proof.
A massive improvement was achieved by relying less on trace lemmas for inductive reasoning but leveraging inductive reasoning in the underlying first-order theorem prover Vampire. More details on recent developments are summarized below.

2. Status

2.1. Lemmaless Induction in Trace Logic

We adjusted our Rapid Verification Framework by adding trace logic specific inductive inferences to the underlying theorem prover Vampire.
Specifically we identified two versions of inductive inferences in the superposition calculus that allow to automatically find induction hypotheses needed to prove properties over loops and unbounded data structures such as arrays.

This work has been tested with great results, substantially improving the number of benchmarks that can be automatically verified with our approach. Further this approach lays the groundwork for future extensions to the Rapid Verification Framework as it allows us to handle inductive reasoning automatically in the prover, rather than having to manually identify trace lemmas.

Our paper on this research has been peer-reviewed and accepted for publication at CICM'22 (Conference on Intelligent Computer Mathematics) and will be presented there in September 2022.

2.2. Invariant Generation and Tool Description

Further we extended the Rapid Framework with a new approach to synthesise inductive loop invariants by leveraging the symbol elimination approach in first-order theorem provers. Essentially, this works by deriving logical consequences from

program semantics and trace lemmas such that many of these consequences might be inductive invariants.

This allows for two things:
(1) Many verification approaches based on invariant generation are limited when it comes to quantification as most such tools rely on SMT-solving (satisfiability modulo theory). Our recent efforts to extract invariants with first-order reasoning showcase how provers such as Vampire can be leveraged for arbitrarily quantified invariant generation.
(2) Secondly, this functionality enables Rapid to be used in invariant search for other tools, specifically for quantified invariants over loops iterating/containing unbounded data structures.

Further, we finally gathered all new verification modes of the Rapid Framework and compiled them into a tool description paper together with the work on invariant synthesis.

Our paper has been reviewed and accepted at the FMCAD'22 conference (Formal Methods in Computer-Aided Design) and will be presented in October.

2.3. Interprocedural Verification

In the meanwhile, we also continued to work on extending Rapid with function calls for interprocedural verification. We could identify a suitable program semantics in trace logic for (tail-)recursive function calls and could successfully manually identify the necessary inductive hypotheses to prove two examples.

I'm currently working on the implementation of this semantics in the Rapid Verification Framework. Once this implementation phase has been successfully completed, I will move on to identify and equip trace lemma and inbuilt inductive reasoning in the theorem prover to find necessary induction hypotheses automatically which will hopefully allow us to handle recursive function calls within the Rapid Framework.

I'm planning to submit this work in the form of a paper in October 2022.

2.4. Blog Articles

So far, five blog articles have been in the works, two of which are going to be published throughout the summer.
Published articles:
+ What is Software Verification?
+ The Rapid Verification Framework

+ Lemmaless Induction in Trace Logic

To appear in July/August 2022:
* Invariant Generation with the Rapid Framework
* Good News: two peer-reviewed and accepted papers to appear at CICM and FMCAD 2022

3. Summary

While I still stick to the original plan, the timeline has to be adjusted as lemmaless induction was essential, but somewhat impromptu work considering recent developments in inductive reasoning with first-order theorem provers. However, it has to be added that lemmaless induction will allow us to proceed faster when it comes to automated inductive reasoning about function calls or any other extension of the Rapid Verification Framework and reasoning in trace logic.
Thus, the work on function calls has been postponed for some months but should still be accomplished throughout this year. I am planning on compiling this work into a paper to be submitted in fall/winter 2022.
As for the verification of concurrent systems, there might be an opportunity to work with peers on a project applying trace logic to ongoing work at the University of Trento which has a leading department on software verification that is internationally renowned. This collaboration is currently planned for early 2023.
Further, the above mentioned publications will be made available after the conference proceedings have been published.