Property Directed Equivalence via Abstract Simulation for Programmable Logic Controller

(Master thesis)

JAN PHILIPP HAFER

Motivation

Even small code changes in safety-critical control software require to re-prove the program semantics against the specification. One formulation of this problem is to show that unsafe variable states, as defined in the specification, are unreachable. However, the process of automatic creation and solving an accurate program model is costly, because a solver must generate and prove the intermediate facts for all possible variable states.

Thus one wants to reuse the model before the code change to generate and verify a model after the code change. Approaches to this idea can be described as incremental regression verification and one approach is property directed equivalence.

State of the Art

Automatic approaches with relation to the field of Programmable Logic Controller use either loop-unrolling techniques or assume loop-free programs. General automatic approaches on loop-containing programs need to 1. identify syntactically changed and unchanged parts between program versions, 2. use a state representation of both program versions that can be solved in an according logic and 3. find a proof for the new program version by reusing the invariants with proof of the old program.
The first part can be attempted with computing a heuristic on a change impact. Expanding or reducing one program to the other is another approach. For second and third part approaches use either a symbolic or value set variable representation with according tradeoffs in memory or computation effort. The underlying problems are hard, so neither solution is guaranteed to succeed.

Goals

The main goal is to apply an approach by Fedyukovich et al. on Programmable Logic Controller programs with the Arcade framework and z3 as SMT solver. The paper suggests multiple strategies to solve the problem and optimization options.

Planned approach

The approach is applied on Large Block Encoding(LBE) of the Control Flow Automata(CFA). LBE can be understand as composition of loops as edges in the CFA. The symbolic representation for the solver is linear integer arithmetic. The CFA construction and LBE transformation is done in Arcade and z3 is used as solver.
First, the node structure of the old program version X is extended to the node structure of the new program version Y. Second, a variable mapping from X to Y is searched via abstracting X towards Y. Third, the proof of X is adapted to find a proof for Y. Similarities of variable names with types and the proof of X can be used for the second step.

Betreuer: Thomas Henn
03.05.2021

Informatik 11
Embedded Software