Accurate Dead Code Detection in Embedded C Code by Arithmetic Constraint Solving*

(Extended Abstract)

Felix Neubauer[1], Karsten Scheibler[1], Bernd Becker[1], Ahmed Mahdi[2], Martin Fränzle[2], Tino Teige[3], Tom Bienmüller[3], Detlef Fehrer[4]

[1] University of Freiburg – `{neubauef|scheibler|becker}@informatik.uni-freiburg.de`
[2] University of Oldenburg – `{mahdi|fraenzle}@informatik.uni-oldenburg.de`
[3] BTC Embedded Systems AG, Oldenburg – `{teige|bienmueller}@btc-es.de`
[4] Sick AG, Waldkirch – `detlef.fehrer@sick.de`

Unreachable code fragments in software, despite not having a negative impact on the functional behavior, can have a bad effect in other areas, such as code optimization or coverage-based code validation and certification. Especially the latter is important in industrial, safety critical environments, where detecting such *dead* code is a major goal to adjust the coverage of software tests.

In the context of embedded systems we focus on C programs which are reactive, control-oriented, and floating-point dominated. Such programs are either automatically generated from Simulink-plus-Stateflow models or hand-written according to coding guidelines. The AVACS transfer project *DeCoDe*, which is a transregional collaboration between the universities of *Freiburg* and *Oldenburg* and the industrial partners *BTC Embedded Systems AG* in Oldenburg and *Sick AG* in Waldkirch aims at finding an accurate and efficient dead-code detection method.

In [2] we describe the current status of the project, where we focus on detecting dead code in automatically generated production C code provided by BTC Embedded Systems. The basic idea is to handle the dead-code detection problem as a code coverage problem, which means that a certain coverage goal, e.g. a statement, is unreachable if and only if it cannot be covered by code coverage analysis. We use a model checking-based tool chain for structural code coverage analysis provided by BTC Embedded Systems to define the coverage goals, perform some code transformations (e.g. cone of influence reduction, loop unwinding, static single assignment form) and translate the code to one SMT formula $\Phi(k)$ with depth k for each coverage goal. This formula is solved by a backend SMT solver, where unsatisfiablity proves depth-bounded unreachability of the coverage goal. Depth-unbounded certificates are obtained by related SMT-based proof schemes, exploiting combinations of k-induction, Craig interpolation, and CEGAR.

We use the SMT solver iSAT3, which is based on interval constraint propagation. It is originally a solver on real arithmetics. Thus a crucial part of our work was adding floating-point support to iSAT3 to ensure accurate reasoning in this domain [3]. In comparison to the state-of-the-art

*This work has been supported by the German Research Foundation (DFG) as part of the Transregional Collaborative Research Center "Automatic Verification and Analysis of Complex Systems" (DFG, SFB/TR 14 AVACS).

bit-blasting solver CBMC (using k-induction) iSAT3 (using Craig interpolation) was able to prove more lines of code to be dead.

There is also some ongoing work regarding the handling of the large state space of the dead-code detection task by using a CEGAR approach [1]. Furthermore we consider hand-written C code, which is highly interrupt-driven and thus cannot be handled by the current method. The support of this code domain – as a mandatory task of the transfer project – is one of our future challenges, besides e.g. some solver improvements.

References

[1] Mahdi, Ahmed, Karsten Scheibler, Felix Neubauer, Martin Fränzle, and Bernd Becker: *Advancing Software Model Checking beyond Linear Arithmetic Theories*. In *Twelfth Haifa Verification Conference - HVC 2016*. Springer, 2016.

[2] Neubauer, Felix, Karsten Scheibler, Bernd Becker, Ahmed Mahdi, Martin Fränzle, Tino Teige, Tom Bienmüller, and Detlef Fehrer: *Accurate Dead Code Detection in Embedded C Code by Arithmetic Constraint Solving*. In *First International Workshop on Satisfiability Checking and Symbolic Computation - FETOPEN-CSA SC2 Workshop 2016*. IEEE, 2016.

[3] Scheibler, Karsten, Felix Neubauer, Ahmed Mahdi, Martin Fränzle, Tino Teige, Tom Bienmüller, Detlef Fehrer, and Bernd Becker: *Accurate ICP-based Floating-Point Reasoning*. In *Formal Methods in Computer-Aided Design, FMCAD 2016*. IEEE, 2016.