Frama-C Exploration

Tripti Agarwal (u1319433)

April 2022

1 Introduction

Frama-C is an open-source extensible and collaborative platform dedicated to source-code analysis of C software. With frama-c we can perform security check, verify requirements and guarantee that the C program is correct. It provide different plugins to perform static and dynamic analysis of software. In this project we explored the power of frama-c by using three different plugins, i.e. wp, wp-rte and eva. We also analysed the output produced by frama-c using the combination of different plugins.

2 Background

2.1 Static Analysis

Static analysis is the testing and evaluation of an application by examining the code without executing the application. It examines all possible execution paths and variable values, not just those invoked during execution. Thus static analysis can reveal errors that may not manifest themselves until weeks, months or years after release. This aspect of static analysis is especially valuable in security assurance, because security attacks often exercise an application in unforeseen and untested ways. Dynamic analysis is the testing and evaluation of an application during runtime.

2.2 Hoare's Logic

Hoare Logic is a formal system with a set of logical rules for reasoning rigorously about the correctness of computer programs. The main feature of Hoare Logic is Hoare's Triple which describes how the execution of a piece of code changes the state of the computation.

$$\{\mathbf{P}\}C\{\mathbf{Q}\} \tag{1}$$

where **P** and **Q** are assertions and **C** is a command. **P** is named the precondition and **Q** the postcondition: when the precondition is met, executing the command establishes the postcondition.

3 Frama-C Plugins' Exploration

3.1 wp

This plugin is an implementation of weakest precondition calculation for annotated C programs. By default, the WP plug-in does not generate any proof obligation for verifying the absence of runtime errors in the code and assumes that the code is runtime error free. A weakest precondition is usually given in form of Hoare triple.

$$\{wp(stmtQ)\}stmt\{Q\} \tag{2}$$

The weakest precondition of property Q through statement stmt should also ensure termination and execution without runtime errors. For this we use another plugin i.e. wp-rte.

3.2 wp-rte

A runtime error is a usually fatal problem encountered when a program is executed. Typical fatal problems are segmentation faults (the program tries to access memory that it is not allowed to access) and floating point exceptions (for instance when dividing an integer by zero: despite its name, this exception does not only occur when dealing with floating point arithmetic). The aim of the RTE plug-in is to automatically generate annotations for common runtime errors, such as division by zero, signed integer overflow or invalid memory access; unsigned integer overflows, which are allowed by the C language but may pose problem to solvers;

3.3 Evolved Value Analysis (Eva)

The Eva plug-in automatically computes sets of possible values for the variables of an analyzed program. Eva warns about possible run-time errors in the analyzed program. Lastly, synthetic information about each analyzed function can be computed automatically from the values provided by Eva.

4 Frama-C commonly used keywords

- Preconsition is given by requires and postcondition is given by ensures.
- assert is used for validation of predicates. Predicates are given using predicate keyword.
- Programs with different results can be given using behaviors. These behaviors can be complete and disjoint or both based on the input it takes and the output produced.
- assigns keyword is given to the variables whoes value is changing in the function.
- Pre/Old : value before function call
- Post : value after function call
- LoopEntry : value at loop entry
- LoopCurrent : value at the beginning of the current step of the loop
- Here : value at current program point
- Other keywords include forall, exists, valid, result, nothing, loop invariant, loop assign, loop variant, etc

5 Codes

- We have written different C codes and performed a proof on them using different frama-c plugins and keywords. Some of the main codes to look at are Inverse Square root, Binary Search, Quick Sort, Stable Quicksort.
- To check the power of frama-c we also manually inserted some bugs in the code. Frama-c was successfully able to identify that the assertions and loop invariants are not satisfied in the broken code. The broken code can be found here.
- We also checked the code by using klee assertion, to see if the code works correctly or not. The link for klee code can be found here.

6 Results

- We observe that the eva plugin can produce values for every variable in the program. We also observed that the eva plugin produce alarms for every potential point of error in the code. Eva can handle integer and floating point numbers. The results for each function using eva plugin are in the form of set of all possible outputs.
- The wp plugin checks whether the code follows Hoare logic. It checks based on certain precondition (requires) when a command is run, the postcondition (ensures) is satisfied. It also checks whether the loop invariant is correct at each point of the loop.
- The wp-rte plugin produces all the runtime error if it is present in the code. For example in the inverse square root code, if the number is 0 then 1/sqrt(0) will produce an error. This type of error can be easily identified by wp-rte. It can also identify if the pointers are pointing to a valid memory location.
- Frama-c also has a GUI which can be used for color coding the statements that are proved or half way proved or invalid, etc. Some commonly used colors are : yellow meaning the property has not been validated, half green and half yellow meaning property is valid but has dependencies, green meaning property and all its dependencies are valid, red meaning invalid property.

- Another notion in frama-c is smoke tests. If the preconditions are inconsistent we can use smoke tests to identify these inconsistencies. Frama-c GUI helps in understanding this easily by color coding the preconditions as red/orange. In this orange means that there exists a reachable call to the function that the precondition will be necessarily violated. The red color indicates that the prover have succeeded in proving that this precondition is inconsistent.
- For more explanation of frama-c results look at the ppt with annotated outputs.

7 Conclusion and Future Work

In conclusion, we can say frama-c is powerful enough to verify C programs. But due to its large documentation it becomes a little difficult to find error in the code. For future work, it will be really interesting to explore frama-c for openmp and cuda programs.

8 References

References

[1] Møller, Anders, and Michael I. Schwartzbach. "Static program analysis." Notes. Feb (2012).

[2] Gordon, Mike. "Background reading on hoare logic." Lecture Notes, April (2012).

[3] https://frama-c.com/

[4] Cuoq, Pascal, Florent Kirchner, Nikolai Kosmatov, Virgile Prevosto, Julien Signoles, and Boris Yakobowski. "Frama-c." In International conference on software engineering and formal methods, pp. 233-247. Springer, Berlin, Heidelberg, 2012.

[5] Certezeanu, Razvan, Sophia Drossopoulou, Benjamin Egelund-Muller, K. Rustan M. Leino, Sinduran Sivarajan, and Mark Wheelhouse. "Quicksort Revisited." In Theory and Practice of Formal Methods, pp. 407-426. Springer, Cham, 2016.