Karlsruhe Institute of Technology

Formal Systems II: Applications

Bounded Model Checking of C Programs and LLBMC

Carsten Sinz | SS 2019

Institute for Theoretical Computer Science
Research Group “Verification meets Algorithm Engineering”

Exercise 1: Get to Know LLBMC

- Using LLBMC is two-step process
  1. Compile C program to LLVM-IR (bitcode)
  2. Run LLBMC on bitcode
- Example:
  - `clang -c -g -emit-llvm abs.c`
  - `llbmc abs.bc`
- Check options:
  - `llbmc -help`
  - `llbmc -help-hidden`
- Make some experiments:
  - Switch on/off check options
  - Display intermediate files
  - Display counterexample
  - Check `llbmc.h`

```
int abs(int x)
{
    int ret;
    if (x >= 0) {
        ret = x;
    } else {
        ret = -x;
    }
    __llbmc_assert(ret >= 0);
    return ret;
}
```

Exercise 2: Triangle Classification

- Download the file `https://baldur.iti.kit.edu/llbmc-ex.zip`
- Unzip the file, containing C source for classification of triangles
  - Triangles are determined by 3 points in 2D space, given by (x,y)-coordinates
  - The x and y coordinates are integers
- The task is to write:
  - A) Functions to classify triangles by angles and sides (see `triangle.h`)
  - B) Write test cases for A)

Exercise 2: Group A

- Write code to classify triangles by angles and sides.
- Check your implementation for run-time errors with LLBMC.

Exercise 2: Group B

- Write test cases for the triangle classification code.
- Also write „generalized tests“, summarizing a set of test cases, using LLBMC‘s `__llbmc_assume()` and `__llbmc_assert()` functions.