CS569 HW1
Jingwei Zhang
Email: zhangj5@onid.oregonstate.edu

The CBMC that installed in my laptop is version 5.0. It runs in my laptop successfully. See figure 1.

```
10-162-151-134:cs569hw1 element$ cbmc test1.c --function sum
CBMC version 5.0 64-bit macos
Parsing test1.c
Converting
Type-checking test1
Generating GOTO Program
Adding CPROVER library
Function Pointer Removal
Partial Inlining
Generic Property Instrumentation
Starting Bounded Model Checking
Unwinding loop sum.0 iteration 1 file test1.c line 7 function sum thread 0
Unwinding loop sum.0 iteration 2 file test1.c line 7 function sum thread 0
Unwinding loop sum.0 iteration 3 file test1.c line 7 function sum thread 0
Unwinding loop sum.0 iteration 4 file test1.c line 7 function sum thread 0
Unwinding loop sum.0 iteration 5 file test1.c line 7 function sum thread 0
Unwinding loop sum.0 iteration 6 file test1.c line 7 function sum thread 0
Unwinding loop sum.0 iteration 7 file test1.c line 7 function sum thread 0
Unwinding loop sum.0 iteration 8 file test1.c line 7 function sum thread 0
Unwinding loop sum.0 iteration 9 file test1.c line 7 function sum thread 0
Unwinding loop sum.0 iteration 10 file test1.c line 7 function sum thread 0
size of program expression: 66 steps
simple slicing removed 0 assignments
Generated 0 VCC(s), 0 remaining after simplification
VERIFICATION SUCCESSFUL
```

Figure 1: the result of using cbmc to verify a function

binarysearch.c is source code that was derived from http://www.programmingsimplified.com/c/source-code/c-program-binary-search. As far as I have known, there are three bugs in the source code. harness_hw1.c is used to look for debugging binarysearch.c. Type command: cbmc binarysearch.c harness_hw1.c –DSIZE = 10 --unwind 10 --bounds-check then get figure 2.

```
Unwinding loop main.1 iteration 1 (12 max) file harness_hw1.c line 29 function main thread 0
Unwinding loop main.1 iteration 2 (12 max) file harness_hw1.c line 29 function main thread 0
Unwinding loop main.1 iteration 3 (12 max) file harness_hw1.c line 29 function main thread 0
Unwinding loop main.1 iteration 4 (12 max) file harness_hw1.c line 29 function main thread 0
Unwinding loop main.1 iteration 5 (12 max) file harness_hw1.c line 29 function main thread 0
Unwinding loop main.1 iteration 6 (12 max) file harness_hw1.c line 29 function main thread 0
Unwinding loop main.1 iteration 7 (12 max) file harness_hw1.c line 29 function main thread 0
Unwinding loop main.1 iteration 8 (12 max) file harness_hw1.c line 29 function main thread 0
Unwinding loop main.1 iteration 9 (12 max) file harness_hw1.c line 29 function main thread 0
Unwinding loop main.1 iteration 10 (12 max) file harness_hw1.c line 29 function main thread 0
Unwinding loop main.1 iteration 11 (12 max) file harness_hw1.c line 29 function main thread 0
Not unwinding loop main.1 iteration 12 (12 max) file harness_hw1.c line 29 function main thread 0
Unwinding loop main.2 iteration 11 (12 max) file harness_hw1.c line 28 function main thread 0
Unwinding loop main.1 iteration 1 (12 max) file harness_hw1.c line 29 function main thread 0
Unwinding loop main.1 iteration 2 (12 max) file harness_hw1.c line 29 function main thread 0
Unwinding loop main.1 iteration 3 (12 max) file harness_hw1.c line 29 function main thread 0
Unwinding loop main.1 iteration 4 (12 max) file harness_hw1.c line 29 function main thread 0
Unwinding loop main.1 iteration 5 (12 max) file harness_hw1.c line 29 function main thread 0
Unwinding loop main.1 iteration 6 (12 max) file harness_hw1.c line 29 function main thread 0
Unwinding loop main.1 iteration 7 (12 max) file harness_hw1.c line 29 function main thread 0
Unwinding loop main.1 iteration 8 (12 max) file harness_hw1.c line 29 function main thread 0
Unwinding loop main.1 iteration 9 (12 max) file harness_hw1.c line 29 function main thread 0
Unwinding loop main.1 iteration 10 (12 max) file harness_hw1.c line 29 function main thread 0
Unwinding loop main.1 iteration 11 (12 max) file harness_hw1.c line 29 function main thread 0
Not unwinding loop main.1 iteration 12 (12 max) file harness_hw1.c line 29 function main thread 0
Not unwinding loop main.2 iteration 12 (12 max) file harness_hw1.c line 28 function main thread 0
Unwinding loop main.3 iteration 1 (12 max) file harness_hw1.c line 39 function main thread 0
Unwinding loop main.3 iteration 2 (12 max) file harness_hw1.c line 39 function main thread 0
Unwinding loop main.3 iteration 3 (12 max) file harness_hw1.c line 39 function main thread 0
Unwinding loop main.3 iteration 4 (12 max) file harness_hw1.c line 39 function main thread 0
Unwinding loop main.3 iteration 5 (12 max) file harness_hw1.c line 39 function main thread 0
Unwinding loop main.3 iteration 6 (12 max) file harness_hw1.c line 39 function main thread 0
Unwinding loop main.3 iteration 7 (12 max) file harness_hw1.c line 39 function main thread 0
Unwinding loop main.3 iteration 8 (12 max) file harness_hw1.c line 39 function main thread 0
Unwinding loop main.3 iteration 9 (12 max) file harness_hw1.c line 39 function main thread 0
Unwinding loop main.3 iteration 10 (12 max) file harness_hw1.c line 39 function main thread 0
Unwinding loop main.3 iteration 11 (12 max) file harness_hw1.c line 39 function main thread 0
Not unwinding loop main.3 iteration 12 (12 max) file harness_hw1.c line 39 function main thread 0
**** WARNING: no body for function binarysearch
size of program expression: 2731 steps
simple slicing removed 31 assignments
Generated 639 VCC(s), 132 remaining after simplification
Passing problem to propositional reduction
Running propositional reduction
Post-processing
Solving with MiniSAT 2.2.0 with simplifier
55638 variables, 204630 clauses
SAT checker: negated claim is UNSATISFIABLE, i.e., holds
Runtime decision procedure: 0.615s
VERIFICATION SUCCESSFUL
10-249-118-63:cs569hw1 element$
```

Figure 2: The result after type: cbmc binarysearch.c harness_hw1.c --DISIZE=10 --unwind 14

According to figure 2, it seems like the source code is correct because the verification is successful. However, if using compiler gcc to verify the source code, it is buggy. See figure 3:

```
10-162-151-134:cs569hw1 element$ ./binarysearch
Enter number of elements
8
Enter sorted 8 integers
1 2 3 4 5 6 7 8
Enter value to find
9
```

Figure 3: The result of testing source code using gcc compiler

This unexpected result is due to line 53 and line 63 from binarysearch.c. For example, assuming array = [1,2,3,4,5,6,7,8], search number is 100. Then after several binary search, the variable first equals to 6 and variable last equals to 7 all the time. In this case the while loop will run forever. From my opinion, the reason why harness_hw1.c program can't detect this error is because harness_hw1.c doesn't verify the left number of numbers need to be checked in the array will decrease when binary search runs every time. Also harness.c doesn't check the validity of number of integers entered into program by users. In some cases, users may type more number of integers than his expected. But this type error may not cause any false result because program will ignore the redundant integers.
There is any bug lies in line 46 in binarysearch.c. Variable last is the position of the last integer in array. Because position starts at 0, so the last integer should at n – 1. Actually it's difficult to detect this error because in many cases there is no effect caused by this error at all. I don't know how to write code to verify the validity of the boundary of data entered by users. But I think harness could find this bug because cmbc is capable of looking for errors related to array bounds after --bounds-check was add into command.

According to testing result, the running time of decision procedure will increased when the number of SIZE in harness_hw1.c goes up and the number of unwind increased. We should note that –unwind must be large enough to cover all program paths. If this prerequisite is not satisfied, the verification of harness_hw1.c may failed. See Figure 4:

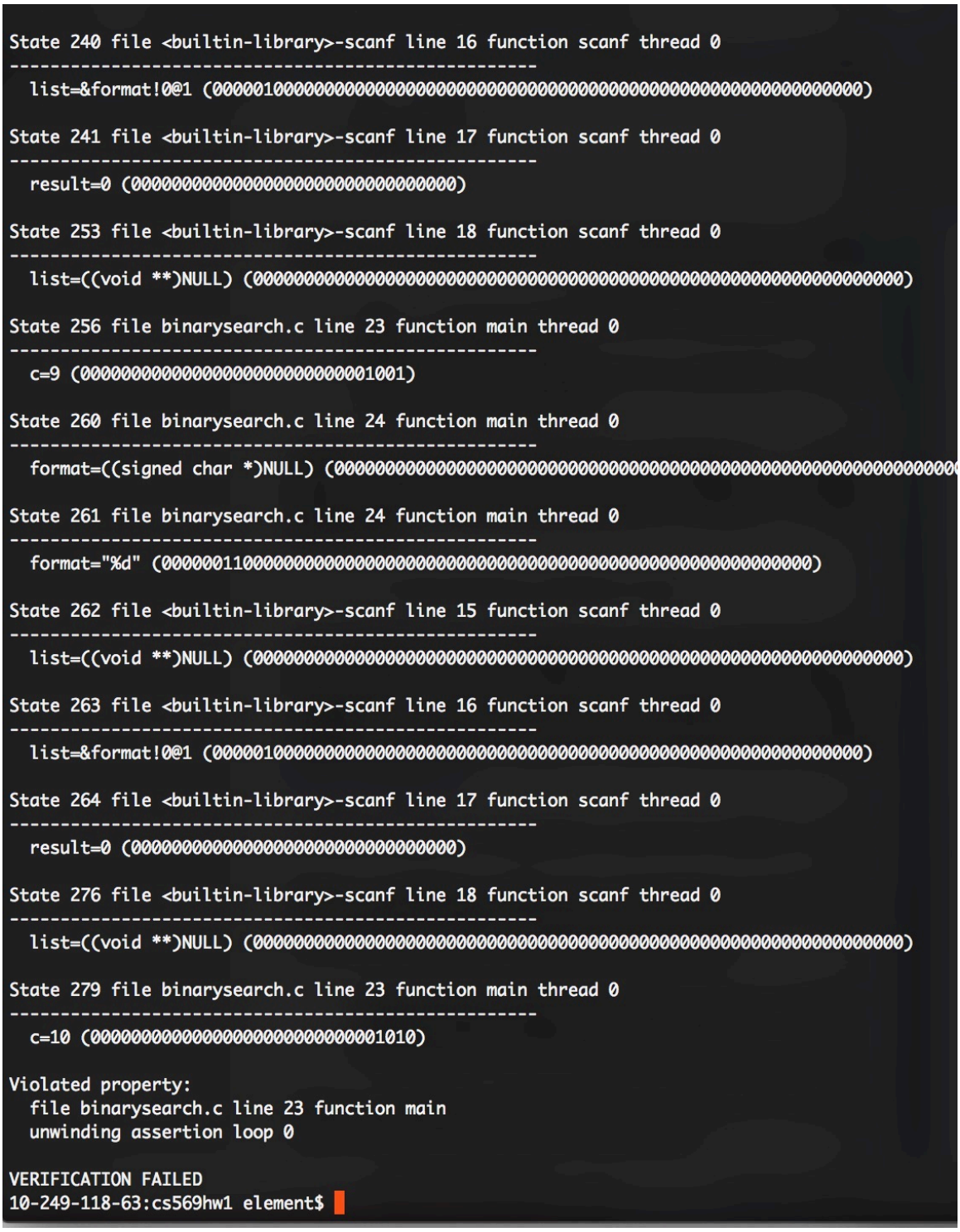

Figure 4: The result of verification when number of --unwind is not large enough

In conclusion, the source code is incorrect. But it's difficult to verify the correctness of source code with harness_hw1.c.