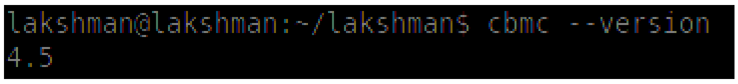
Lakshman Madhav Kollipara
CS569_Assignment 1

Assignment 1
CS569_Static Analysis and Model checking
Alex Groce

What is CBMC?
C- Bounded Model Checking.
Programs written in C do not provide automatic bounds checking on the buffer, which means a program can – accidentally or maliciously – write past a buffer. Such static errors are identified by Bounded model checking. CBMC is one of the Static Analysis tool where we analyze and optimize the code without running it. CBMC satisfies 100% completeness which means it tells about all the bugs (May be a real bug sometimes may not).

Installing CBMC:
I have installed latest version (4.5) of CBMC from cprover.org.



```
lakshman@lakshman:~/lakshmans cbmc --version
4.5
```

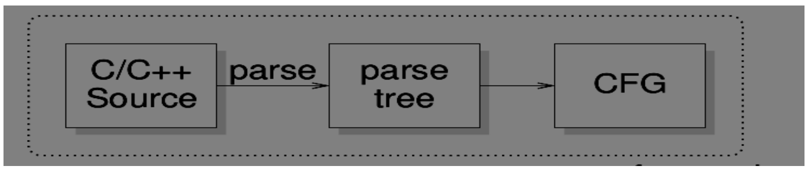
C program Tested & Harness:
Taking 2 integer arrays as input, merging them into one array and sorting the array.
Checking the correctness of output by following 3 assertions:

- Asserting if array 1 is subset of Output.
- Asserting if array 2 is subset of output.
- Asserting if the final output is sorted or not (by incrementing count if a[1]>a[0])
- Asserting if size of output is equal to size of input

I implemented 3 functions: one for merging two arrays, one for sorting, and one for checking subset. My first version of program has too many bugs, which I debugged using asserts and believe that my program is reliable up to some extent.

I have included my test suit (so called harness) to input 2 arrays and then calling my merge and sort and test the thoroughness of output using asserts.

Working with CBMC:
CBMC is a Bounded Model Checker for C and C++ programs. It allows verifying array bounds (buffer overflows), pointer safety, exceptions and user-specified assertions. The verification is performed by unwinding the loops in the program and passing the resulting equation to a decision procedure.



What I understood from working of CBMC is, it parses the C-program and converts into Control flow graph, where the graph is traversed and a set of constraints are generated along with if any user specified constraints (asserts and assumes) and converts into a set of Boolean expressions. These expressions are passed to a SAT solver and tries to solve the equation. If the conditions are unsatisfiable, then it gives a counter example. If there are multiple variable assignments, it uses versioning and gives different representations to each use of variable.

Lakshman Madhav Kollipara
CS569_Assignment 1

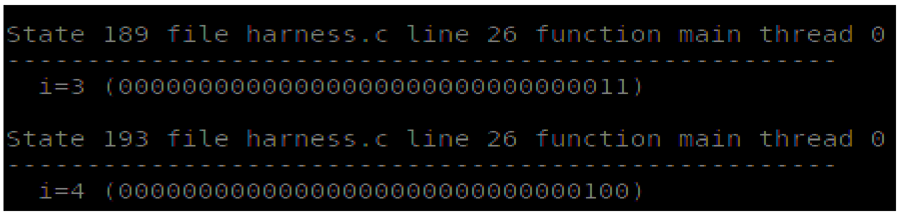
How to run CBMC:
cbmc - -help → to check the usage of available options
cbmc filename.c →To run cbmc on specified file.
cbmc filename.c - -bounds-check → Checks whether the array is out of the bound or not.
cbmc filename.c - -pointer-check → To verify whether the pointer de-referencing has any errors or not.
cbmc filename.c - -unwind v→ Unwinds the loops in program 'v' times.
cbmc filename.c - -no-unwinding-assertions → if the unwinding value we specified is less than the loop actual run times, then ignores loop unwinding failed errors.
cbmc filename.c - - function func_name → we can specify entry point to specific function in our program.
Cbmc filename.c - -show-claims → Prints the list of properties it is checking.
CBMC semantics:
int nondet_int(); → generates any random integer value restricted by using assume.
__CPROVER_assume() → limiting the range of random numbers chosen by nondet_int()

If the program is verified successfully then it just prints VERIFICATION SUCCESSFUL with the number of SAT variables and clauses created and overall run time.
If the program failed to verify i.e if it's UNSAT, it gives a counter example and gives which property is failing.

Running CBMC on my program:
Command to run cbmc on my harness: cbmc harness.c cbmc.c -DSIZE=3 –unwind 7 –bounds-check

When I first ran cbmc on my harness, I ended up in getting verification failed. So, I tried debugging till I get verification successful. And then I tried introducing a minor bug in my C code and then analyzing how cbmc is useful. First if I change something in my code, my harness is detecting it and verification failed because, one of the assertion fails. The CBMC is giving me which assertion fails and why i.e at which state the value on LHS is not equal to RHS and how the variables are changing in each state. Using printf statements, the debugging will be much easier, because instead of back tracking all the states, we can see the LOG. One thing I didn't get here is CBMC is not actually running the program. But, how will it assign variables, perform operation and print statement only if verification failed but not when verification successful. And one more doubt is why some states are missing when I look at the states printing when verification is failed, I can see only few states but not all.

Refer to figure below.


```
State 189 file harness.c line 26 function main thread 0
----------------------------------------------------
  i=3 (00000000000000000000000000000011)

State 193 file harness.c line 26 function main thread 0
----------------------------------------------------
  i=4 (00000000000000000000000000000100)
```

Here what about States 190,191,192 ??

If I try assigning a value to array element which is greater than upper bound, If I didn't specify –bounds-check while running cbmc, it prints verification successful which is not really successful. So, whenever we are using array or pointers in our code, we should use –bounds-check and –pointer-check.

The interesting part is when I run code without bounds check it is taking 257.64s to run. Whereas running with bounds check is taking 255.549s to run. I expected cbmc to take more time with bounds check but it’s vice versa.

You may wonder why my program is taking so long to run because, my code has too many loops where I am going through the whole array which increases overhead. I met you regarding this and you suggested me to check isSubset in different way, but u had already discussed it in the class. So I am running with my function which checks whether each element is subset or not. I have made few changes on assume as you said not to restrict everything. And I removed few un-necessary for loops and try to reduce the overhead but not completely.

My program runs in ~3 seconds for Array size 2 and Unwinding 5 and ~300 seconds for size 3 and unwind 7 which increased quite a lot.

I have one more question, why the number of variables and clauses are increasing when we include a printf statement? Does cbmc works with character inputs and strings? I tried running with characters and when I try to print, its printing some random integer values.

The harder part for me in this course is, As I have done CS562 last term I am looking testing only in perspective of correctness of output and effective asserts. I sometimes find hard to think in cbmc way and do some static analysis. Even when writing the code, my intention was always focusing on getting correct output and checking it using asserts. So my code will look good for dynamic analysis. I hope by end of this course I will write code which is compatible and much usable to test with cbmc. I didnt try with any pointer check is because, I didnt understand what exactly pointer-check does.

I liked the way which you tested the correctness of sorted array by taking refc (count of value in input) and acount (count of value in output) and comparing them. I tried even including that here but as my run time is too high I didn't include it.

I accept that my report is too long. But the first 3 sections are for my clarification of what I learnt and last section explains pretty well about assignment.

Submitted By:
Lakshman Madhav Kollipara
For CS569_Assignment 1
For CS569_Assignment 1