DD2459 Software Reliability
Lab 1

Yuze Cui
yuzec@kth.se
Rui Shi
srui@kth.se

Lab 1: White-box Testing

March 20, 2022

1 Draw a condensation graph for the Triangle Test algorithm.

1.1

(a) Write a set of test requirements that achieve full node coverage (NC) for the Triangle Test algorithm.

Answer:
NC TR1: n_1, n_{10}
NC TR2: n_1, n_2, n_{20}
NC TR3: n_1, n_2, n_3, n_{30}
NC TR4: n_1, n_2, n_3, n_4, n_{40}
NC TR5: n_1, n_2, n_3, n_4, n_{41}

(b) Write out a minimized set of test cases satisfying the requirements of (a).
Answer:
triangleTest(-1, 2, 3) - TR1
triangleTest(1, 2, 3) - TR2
triangleTest(2, 2, 2) - TR3
triangleTest(2, 2, 3) - TR4
triangleTest(2, 3, 4) - TR5

1.2

(a) Write out a set of test requirements that achieve full edge coverage (EC) for the Triangle Test algorithm.

Answer:
The requirements are the same as Node Coverage:
EC TR1: n_1, n_{10}
EC TR2: n_1, n_2, n_{20}
EC TR3: n_1, n_2, n_3, n_{30}
EC TR4: n_1, n_2, n_3, n_4, n_{40}
EC TR5: n_1, n_2, n_3, n_4, n_{41}

(b) Write out a minimized corresponding set of test cases.
Answer:
The edge coverage test cases are the same as Node Coverage:
triangleTest(-1, 2, 3) - TR1
triangleTest(1, 2, 3) - TR2
triangleTest(2, 2, 2) - TR3
triangleTest(2, 2, 3) - TR4

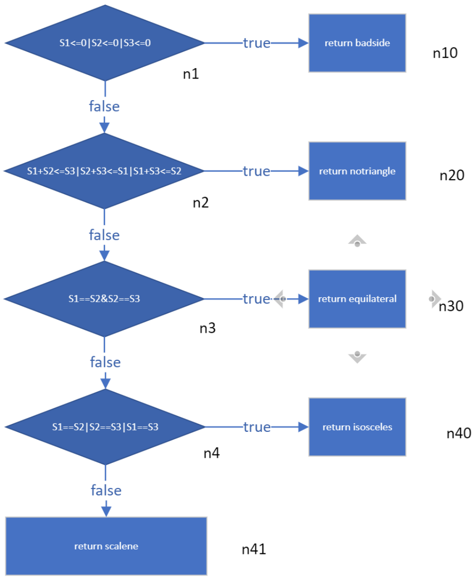

Figure 1: Condensation graph for Q1

triangleTest(2, 3, 4) - TR5

(c) Why are node coverage and edge coverage the same in this example? Carefully explain your reasoning about this fact.

Answer: Because for this example, when every edge is visited, all the node will be visited. And vice versa. That is because each test requirement has different end node to reach.

2 Write out test requirements as logical constraints on the input variable values s1, s2 and s3 to achieve different levels of logic coverage.

2.1

(a) Write out a set of test requirements that achieve full predicate coverage (PC) for the Triangle test algorithm 1. (Recall that non-distributive predicate coverage is sufficient here.) Write the corresponding set of test cases.
Answer:
TR1:S1$\leq$ $0||S2 \leq 0||S3 \leq 0$;
TR2:!(S1$\leq$ $0||S2 \leq 0||S3 \leq 0$);
TR3:S1+S2$\leq$ $S3||S1 + S3 \leq S2||S2 + S3 \leq S1$;
TR4:!(S1+S2$\leq$ $S3||S1 + S3 \leq S2||S2 + S3 \leq S1$);
TR5:$S1 == S2\&\&S2 == S3$;
TR6:!($S1 == S2\&\&S2 == S3$);
TR7:$S1 == S2||S2 == S3||S1 == S3$;
TR8:!($S1 == S2||S2 == S3||S1 == S3$);
Test cases:
PC TC1:triangleTest(-1,1,3);
PC TC2:triangleTest(1,1,3);
PC TR3:triangleTest(3,1,3);
PC TR4:triangleTest(3,1,3);
PC TR5:triangleTest(4,5,3);

(b) Looking back on your answers to 1.1.(b) node coverage (NC) and 2.1.(a) predicate coverage (PC) are these always the same for every condensation graph?
Answer:

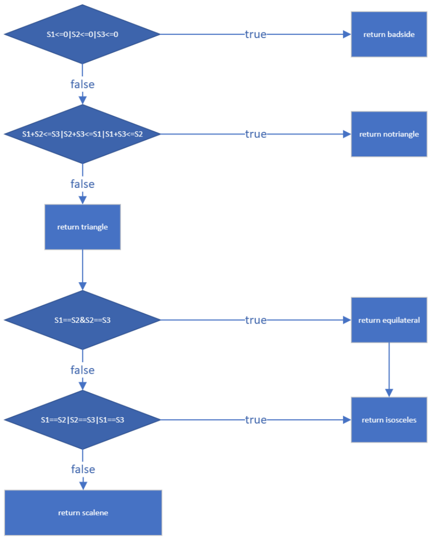

Figure 2: Condensation graph for Q2

No. The node coverage and the predicate coverage are sometimes different. The predicate coverage should include false and true situation. Achieving node coverage doesn't mean visiting all predicate(truefalse), vice versa.

(c) Can you modify your condensation graph for Question 1 in some simple way so that predicate coverage PC and node coverage NC are not the same. You do not have to preserve the functionality of the program. Verify that your answer is correct by writing out corresponding test suites for your PC and NC requirements that are different.
Answer:
Figure2 shows the modified condensation gragh.
For this gragh, we have these test cases below:
PC TC1:S1=−1,S2=1,S3=3;
PC TC2:S1=3,S2=2,S3=1;
PC TC3:S1=2,S2=2,S3=2;
PC TC4:S1=2,S2=2,S3=1;
PC TC5:S1=3,S2=4,S3=5.
For the predicate coverage,the test suite should be {TC1,TC2,TC3,TC4,TC5},however, full node coverage need the test suite of{TC1,TC2,TC3,TC5}.

2.2

(a) Write out a set of test requirements that achieve full clause coverage (CC) for the Triangle Test Algorithm, using your condensation graph model.
Answer:
CC TR1:S1$\leq$ 0;
$CCTR2 : S2 \leq 0;$
$CCTR3 : S3 \leq 0;$
$CCTR4 : S1 > 0;$
$CCTR5 : S2 > 0;$
$CCTR6 : S3 > 0;$

CC TR7:S1+S2$\leq S3$;
$CCTR8 : S3 + S2 \leq S1;$
$CCTR9 : S1 + S3 \leq S2;$
$CCTR10 : S1 + S2 > S3;$
$CCTR11 : S3 + S2 > S1;$

$CCTR12 : S1 + S3 > S2;$

CC TR13:S1==S2;
CC TR14:S2==S3;
CC TR15:S1==S3;
CC TR16:S1!=S2;
CC TR17:S2!=S3;
CC TR18:S1!=S3;

(b) Write out a corresponding set of test cases.
Answer:
CC TC1:S1=1,S2=1,S3=1;(Covers TR13,TR14,TR10,TR11,TR12)
CC TC2:S1=0,S2=0,S3=0;(Covers TR1,TR2,TR3)
CC TC3:S1=1,S2=1,S3=2;(Covers TR4,TR5,TR6,TR7)
CC TC4:S1=2,S2=1,S3=1;(Covers TR8)
CC TC5:S1=1,S2=2,S3=1;(Covers TR9)
CC TC6:S1=3,S2=4,S3=5;(Covers TR16,TR17,TR18)
CC TC7:S1=2,S2=1,S3=2:(Covers TR15)

2.3

(a) Write out a set of test requirements that achieve full restricted active clause coverage (RACC) (also known as unique cause MCDC) for the Triangle Test Algorithm, using your condensation graph model
Answer:
To achieve the active clause coverage, we have test requirements for the predicate below:
RACC TR1: S1>0&&S2>0&&S3>0; Predicate1:false
RACC TR2: S1$\leq 0\&\&S2 > 0\&\&S3 > 0;$
$RACCTR3 : S1 > 0\&\&S2 \leq S3\&\&S3 > 0;$
$RACCTR4 : S1 > 0\&\&S2 > 0\&\&S3 \leq 0;$

RACC TR5: S1+S2>S3&&S2+S3>S1&&S1+S3>S2; Predicate2:false
RACC TR6: S1+S2$\leq S3\&\&S2 + S3 > S1\&\&S1 + S3 > S2;$
$RACCTR7 : S1 + S2 > S3\&\&S2 + S3 \leq S1\&\&S1 + S3 > S2;$
$RACCTR8 : S1 + S2 > S3\&\&S2 + S3 > S1\&\&S1 + S3 \leq S2;$

RACC TR9: S1==S2 && S2==S3; Predicate3:true
RACC TR10: S1!=S2 && S2==S3;
RACC TR11: S1==S2 && S2!=S3;

RACC TR12:S1!=S2&&S2!=S3&&S1!=S3; Predicate4:true
RACC TR13:S1==S2&&S2!=S3&&S1!=S3;
RACC TR14:S1!=S2&&S2==S3&&S1!=S3;
RACC TR15:S1!=S2&&S2!=S3&&S1==S3;

(b) Write out a corresponding set of test cases.
Answer:
RACC TC1:S1=1,S2=1,S3=1 (Covers TR1,TR5,TR9)
RACC TC2:S1=0,S2=1,S3=1 (Covers TR2)
RACC TC3:S1=1,S2=0,S3=1 (Covers TR3)
RACC TC4:S1=1,S2=1,S3=0 (Covers TR4)

RACC TC5:S1=1,S2=1,S3=2 (COvers TR6)
RACC TC6:S1=2,S2=1,S3=1 (COvers TR7)
RACC TC7:S1=1,S2=2,S3=1 (COvers TR8)
RACC TC8:S1=3,S2=5,S3=5 (Covers TR10,TR14)

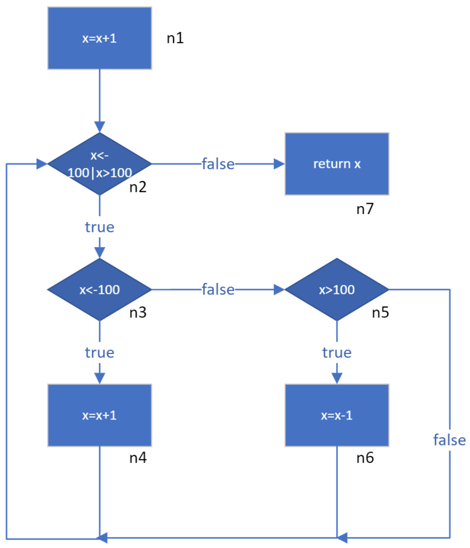

Figure 3: Condensation graph for Q3

RACC TC9:S1=3,S2=3,S3=5 (Covers TR11,TR13)

RACC TC12: S1=3,S2=4,S3=5(Covers TR12)
RACC TC12: S1=5,S2=4,S3=5(Covers TR15)

3 Consider the following piece of code

(a) Draw a condensation graph for this code. Answer:
The answer is figure 3.

(b) Define a minimal set TR of test requirements on the input variable x that would achieve full (100%) node coverage for this program. Carefully explain why your test requirement set is actually minimal.
Answer:
Since the algorithm will limit the input number to a range of from -100 to 100. There should be at least two test cases to achieve full node coverage. Hence, the minimal set can be:
TR1:(n1,n2,n3,n5,n6,n7) (i.e.x>100);
TR2:(n1,n2,n3,n4,n7) (i.e. x<-100);
If the input+1 is above 100, it will decrease the number, which will reach n1,n2,n3,n5,n6, until the input is decrease below 100, then it will reach n7 and exit. However, the n4 cannot be reached if input+1 is smaller than -100, so we should add a input will below -100, so that the TR set can achieve 100% node coverage.

(c) Produce a set TC of test cases that satisfy your test requirements for TR in Part 3.(b).
Answer:
TC1:x=101;
TC2:x=-102;

(d) Would predicate coverage yield a better test suite than your answer to 3.(c)? Motivate your answer.
Answer:
For the predicate coverage, each predicate need to be tested true and false.
We define the predicates:
P1: x$< -100|x >$100;
P2: x<-100;
P3: x>100;

The TC1 x=101 can reach the predicate P1(true&false), P2(false), P3(true); TC2 can reach the predicate P1(truefalse), P2(true). However, the P3 cannot be judged false because after the x=101 is decreased by 1, it will never come back to loop and be determined by P3.
Compared with the test suite in part c, the predicate coverage can never be 100%. So the test suite for PC is not better than the TS of node coverage.

4 Self-Assessment

1. Do you have a test case that represents a valid scalene triangle? NC:yes;
   EC:yes;
   PC:yes;
   CC:yes;
   RACC:yes;

2. Do you have a test case that represents a valid equilateral triangle? NC:yes;
   EC:yes;
   PC:yes;
   CC:yes;
   RACC:yes;

3. Do you have a test case that represents a valid isosceles triangle? NC:yes;
   EC:yes;
   PC:yes;
   CC:yes;
   RACC:yes;

4. Do you have at least three test cases that represent valid isosceles triangles such that you have tried all three permutations of two equal sides? NC:no;
   EC:no;
   PC:no;
   CC:yes;
   RACC:yes;

5. Do you have a test case in which one side has a zero value? NC:no;
   EC:no;
   PC:no;
   CC:yes;
   RACC:yes;

6. Do you have a test case in which one side has a negative value? NC:yes;
   EC:yes;
   PC:yes;
   CC:no;
   RACC:no;

7. Do you have a test case with three integers such that the sum of two is equal to the third? NC:yes;
   EC:yes;
   PC:no;
   CC:yes;
   RACC:yes;

8. Do you have at least three test cases in category 7 such that you have tried all three permutations where the length of one side is equal to the sum of the lengths of the other two sides?

NC:no;
EC:no;
PC:no;
CC:yes;
RACC:yes;

9. Do you have a test case with three integers greater than zero such that the sum of two numbers is less than the third?
NC:no;
EC:no;
PC:yes;
CC:no;
RACC:no;

10. Do you have at least three test cases in category 9 such that you have tried all three permutations NC:no;
EC:no;
PC:no;
CC:no;
RACC:no;

11. Do you have a test case in which all sides are zero? NC:no;
EC:no;
PC:no;
CC:yes;
RACC:no;

12. Do you have at least one test case specifying non-integer values or does this not make sense? NC:no;
EC:no;
PC:no;
CC:no;
RACC:no;

13. Do you have at least one test case specifying the wrong number of values (2 or less, four or more) or does this not make sense?
NC:no;
EC:no;
PC:no;
CC:no;
RACC:no;

14. For each test case, did you specify the expected output from the program in addition to the input values? NC:yes;
EC:yes;
PC:yes;
CC:yes;
RACC:yes;

The points of each test suite can be concluded by the table below:

Coverage	NC	EC	PC	CC	RACC
Points	6	6	6	9	8

Clause coverage achieves the highest score in my table. For the questions above which evaluate the tests, clause coverage test can reach the most circumstance. Hence, for best test performance, CC can be chosen. Additionally, RACC is also a good choice.