1.benchmark

(\* Andrius Andrijauskas (s121060@student.dtu.dk), Lars Bonnicshen (lfbo@dtu.dk) \*)

program

int array[10];

int a;

skip; (\* label 0, a in {0}, array in {0} \*)

a := -1; (\* label 1, a in {-}, array in {0}\*)

while (a <= 10) do (\* label 2, a in {-,0,+}, array in {0, +} \*)

array[a]:= a \* a; (\* label 3, a in {-,0,+}, array in {0, +} a can be -1 and 10 \*)

a := a+1; (\* label 4, a in {-,0,+}, array in {0, +} \*)

od

skip; (\* label 5, a in {+}, array in {0, +} \*)

end

Program graph:

(1,skip;,2), (2,a := -1;,3), (3,a<=10,4), (4,array[a] := a\*a;,5), (5,a := a+1;,3), (3,!a<=10,6), (6,skip;,7)

Detection of signs solutions table 17:

1: a={0} array={0}

2: a={0} array={0}

3: a={-,0,+} array={0,+}

4: a={-,0,+} array={0,+}

5: a={-,0,+} array={0,+}

6: a={+} array={0,+}

7: a={+} array={0,+}

Low boundary violations for array indexing:

(4,array[a] := a\*a;,5),

Good example since before we did not consider that ,array[a] := a\*a; both values ‘a’ are the same and if a={-,0,+} we would get array={-,0,+}. But now it is fixed for the cases where both sides expressions of multiplication or division operation are variables or arrays (e.g. a\*a) and ‘(a+1)\* a’ will already be considered as different disregard to the signs of ‘a’.

2.benchmark

Kamran Manzoor s124569@student.dtu.dk

Noel Vang s082961@student.dtu.dk

Detection of Signs Analysis

program

int buff[5];

int index;

int wlb;

int wub;

index := 4; (label 1)

wlb := -5; (label 2)

wub := 7; (label 3)

while index >= wlb do (label 4)

buff[index] := 10; (label 5)

index := index - 1; (label 6)

od

index := 0; (label 7)

while index <= wub do (label 8)

write buff[index]; (label 9)

index := index+1; (label 10)

od

end

Solution to detection of signs analysis equations is shown below:

A(.) index wlb wub buff

1 {-,0,+} {-,0,+} {-,0,+} {-,0,+}

2 {+} {-,0,+} {-,0,+} {-,0,+}

3 {+} {-} {-,0,+} {-,0,+}

4 {-,0,+} {-} {+} {-,0,+}

5 {-,0,+} {-} {+} {-,0,+}

6 {-,0,+} {-} {+} {+}

7 {-} {-} {+} {-,0,+}

8 {0,+} {-} {+} {-,0,+}

9 {0,+} {-} {+} {-,0,+}

10 {0,+} {-} {+} {-,0,+}

11 {+} {-} {+} {-,0,+}

From the table, we can see that statement with label 5 i.e.,

buff[index] := 10 has a lower bound exception.

Program graph:

(1,index := 4;,2), (2,wlb := -5;,3), (3,wub := 7;,4), (4,index>=wlb,5), (5,buff[index] := 10;,6), (6,index := index-1;,4), (4,!index>=wlb,7), (7,index := 0;,8), (8,index<=wub,9), (9,write buff[index];,10), (10,index := index+1;,8), (8,!index<=wub,11)

Detection of signs solutions table 33:

1: index={0} wub={0} buff={0} wlb={0}

2: index={+} wub={0} buff={0} wlb={0}

3: index={+} wub={0} buff={0} wlb={-}

4: index={-,0,+} wub={+} buff={0,+} wlb={-}

5: index={-,0,+} wub={+} buff={0,+} wlb={-}

6: index={-,0,+} wub={+} buff={0,+} wlb={-}

7: index={-} wub={+} buff={0,+} wlb={-}

8: index={0,+} wub={+} buff={0,+} wlb={-}

9: index={0,+} wub={+} buff={0,+} wlb={-}

10: index={0,+} wub={+} buff={0,+} wlb={-}

11: index={+} wub={+} buff={0,+} wlb={-}

Low boundary violations for array indexing:

(5,buff[index] := 10;,6)

This bench mark is interesting to show that our program correctly finds a violation. However, it seems that results of solution table provided by the authors of benchmark are not correct.

3 benchmark

(\*Tomasz Cezary Maciazek s111954@student.dtu.dk\*)

(\*Arthur Hugo Maxime Desjardins s131187@student.dtu.dk\*)

program

int x;

int y;

int A[5];

x := 3; (\*label 1 x in {0}, y in {0}, A in {0}\*)

y := 2; (\*label 2 x in {+}, y in {0}, A in {0}\*)

while x > 0 do (\*label 3 x in {0,+}, y in {-,0,+}, A in {0,+}\*)

y := y - 1; (\*label 4 x in {+}, y in {-,0,+}, A in {0,+}\*)

x := x - 1; (\*label 5 x in {+}, y in {-,0,+}, A in {0,+}\*)

A[y] := x; (\*label 6 x in {0,+}, y in {-,0,+}, A in {0,+}\*)

A[x] := x + 2; (\*label 7 x in {0,+}, y in {-,0,+}, A in {0,+}\*)

A[5-x] := 2; (\*label 8 x in {0,+}, y in {-,0,+}, A in {0,+}\*)

od

skip; (\*label 9 x in {0}, y in {-,0,+}, A in {0,+}\*)

end

The detection of signs should return an error for labels 6 and 8.

Our:

Program graph:

(1,x := 3;,2), (2,y := 2;,3), (3,x>0,4), (4,y := y-1;,5), (5,x := x-1;,6), (6,A[y] := x;,7), (7,A[x] := x+2;,8), (8,A[5-x] := 2;,3), (3,!x>0,9), (9,skip;,10)

Detection of signs solutions table 24:

1: A={0} y={0} x={0}

2: A={0} y={0} x={+}

3: A={-,0,+} y={-,0,+} x={-,0,+}

4: A={-,0,+} y={-,0,+} x={+}

5: A={-,0,+} y={-,0,+} x={+}

6: A={-,0,+} y={-,0,+} x={-,0,+}

7: A={-,0,+} y={-,0,+} x={-,0,+}

8: A={-,0,+} y={-,0,+} x={-,0,+}

9: A={-,0,+} y={-,0,+} x={-,0}

10: A={-,0,+} y={-,0,+} x={-,0}

Low boundary violations for array indexing:

(6,A[y] := x;,7), (7,A[x] := x+2;,8)

Very good example to show our limitations

We output edge (7,A[x] := x+2;,8) as well since we do not work with values. Even though Boolean condition x>0 reduces signs of x to {+} the assignment x := x - 1; makes signs of x {-,0,+} since {+} – {+} can result in anything.

We do not output (8,A[5-x] := 2;,3) since we do not parse arithmetic expression in array index at the point of finding violations and we just parse number and lookup signs of variables and signs of arrays in the solutions table.