Project 2

Metode formale în ingineria software 2018-2019
Formal Methods in Software Engineering 2018-2019

Deadline: Saturday, January 26, 20:00.
A pdf file including the detailed description of the solution will be uploaded using the link
https://www.dropbox.com/request/jqnq5sx9VGzMrbVPZqF4

Exercise 1 The goal is to do live variable analysis for the following program

```
a = 2;
b = 4;
if ( b < 2 ) {
  while (b < c)
    b = b + 2;
  a = b;
}
else
  d = a - c;
d = b + 2;
```

using the fixed point approach. The variable analysis is described in Lectures 2 and 4 in [1].

1. Build the control flow graph.
2. Define the functions `kill` and `gen`.
3. Define the transfer function.
4. Define the system of equations.
5. Apply the algorithm computing the fixed point (based on Knaster-Tarski Theorem).

Exercise 2 The goal is to do the constant propagation analysis on the following program:

```
if (x < 0) {
  a = 1;
  if (y > 0)
    { b = 2; c = a + b; }
```

```
  else
    { b = 3; c = b - a; }
}
else {
  b = 1; c = 2;
  if (y < 0)
    a = b + c;
  else
    a = 6 - (b+c);
}
```

using MOP approach. The MOP approach and the variable analysis are described in Lectures 5 and 6 in [1].

1. Build the control flow graph.
2. Define the state.
3. Define the transfer function.
4. Apply the MOP algorithm.

Exercise 3 Design a Uninitialized Variable Analysis that determines, at each program point, whether a variable used at that point is not initialized. You may use Constant Propagation Analysis as inspiration source.

1. Define the partial ordered set of the values for variables.
2. Define the analysis information domain $(D, \sqsubseteq)$ and describe how it can be used to decide whether a variable used at a program point is not initialized.
3. Decide whether the analysis is forward or backward.
4. Define the transfer function.
5. Apply it (using the fixed point approach or the MOP approach) on the following program:

```
x = 1;
y = -1;
if (x * y < 0) z = 1;
if (z > 0) y = w;
u = y;
```

6. How precise is your analysis?

References

[1] Thomas Noll. Static program analysis. Software Modeling and Verification Group, RWTH Aachen University. https://moves.rwth-aachen.de/teaching/ws-1617/spa/

Prof. dr. Dorel Lucanu