CS 744 - Advanced Compiler Design
Assignment 1

Before you complete this assignment, I recommend that you read the following paper:

> John B. Kam and Jeffrey D. Ullman. Monotone Data Flow Analysis Frameworks. *Acta Informatica*, 7:305–317, 1977. `http://dx.doi.org/10.1007/BF00290339`

Reading the paper is optional but recommended. I will not ask you to hand in a summary, but you are welcome to discuss the paper with me if you have any questions or comments about it. Note: There are two differences between the presentation in this paper and in class. First, the encoding of dataflow facts as lattice elements is upside-down throughout, with meet substituted for join, $\geq$ for $\leq$, etc. Second, the paper defines the term *bounded* to mean something very different than what we defined it to mean in class.

Consider the following code in a Jimple-like intermediate representation:

```
1  int a, b, c;
2  a = 0;
3  c = 1;
4  L1:
5  b = a+1;
6  c = c+b;
7  a = b*2;
8  if(a < 10) goto L1;
9  return c;
```

1. (3 marks) Draw a control flow graph for the code.
2. (1 mark) Which variables are live between lines 5 and 6?
3. (1 mark) Which variables are live between lines 7 and 8?
4. (12 marks) Define a dataflow analysis that will determine, at each program point, and for each integer variable, whether the values that it holds may be positive, negative, zero, or some combination of these three. Follow the steps presented in class. State any assumptions that you make.

5. (3 marks)

   (a) Is there a program for which your analysis reports that a given variable may be negative, yet the variable never takes on a negative value in any execution of the program? If yes, give an example of such a program.

   (b) Is there a program for which your analysis reports that a given variable may not be negative, yet there is some execution of the program in which the variable takes on a negative value? If yes, give an example of such a program and execution trace.

6. (12 marks) Implement your analysis using Soot. Follow the general pattern of the live variable analysis example presented in class (the code is available on the course web site). You may use either Soot on the command line, or the Soot-Eclipse plugin. Submit your implementation by e-mail to `olhotak@plg`. Include with your submission any test program(s) on which you have tried your analysis. Submit a single `tar`, `jar`, or `zip` file as an e-mail attachment.