Quiz
Module SOS

Authorised documents : course notes.
Duration : 2 hours.
For each question is given an approximate number of points attributed for a correct answer

1 Data Flow Analysis

An expression is said to be *very busy* at a program point if, no matter what execution path is taken from that point, the expression will always be evaluated before any of the variables occurring in the expression are modified.

Question 1.[4pt] The goal is to formalize the Very Busy Expression analysis as a data flow analysis over the set $\mathbf{AExp}_P$ of arithmetic expressions occurring in the program P to analyse.

The analysis should use the abstract domain: $\mathcal{P}(\mathbf{AExp}_P)$.

Answer the following questions and complete the data flow equations below.

- Is it a forward or a backward analysis?
- Are we interested in the greatest or the least solution to the data flow equations?
- Explain why the analysis terminates.

For all labelled program points l of P,

$$VBE_{in}(l) = \ldots$$
$$VBE_{out}(l) = \ldots$$

$$\mathbf{kill}([x := a]^l) = \ldots$$
$$\mathbf{kill}([skip]^l) = \ldots$$
$$\mathbf{kill}([b]^l) = \ldots$$

$$\mathbf{gen}([x := a]^l) = \ldots$$
$$\mathbf{gen}([skip]^l) = \ldots$$
$$\mathbf{gen}([b]^l) = \ldots$$

Question 2.[3pt] Consider the following program

(**while** (a > b) **do** { x := b - a }) ; y := b - a ; x := a - b

Label the program points, generate the set of data flow equations for the Very Busy Expression analysis of this program, and solve the data flow equations.

2 Bytecode verification

For this question, the term "bytecode verifier" refers to the approach "Typing as data flow analysis" defined in course notes on bytecode verification.

We consider the following bytecode program

```
0:  ifle 6
1:  iload 1
2:  iconst 1
3:  iadd
4:  istore 2
5:  goto 9
6:  new Object
7:  astore 2
8:  goto 9
9:  iload 1
```

Question 3.[4pt]

Show that this program is typable by the bytecode verifier with `MaxStack` = 3 and `MaxReg` = 3, starting with an operand stack [`int`] (one element of type `int`) and a register 1 of type `int` .

Question 4.[1pt] What happens if we replace the instruction at line 8 by `iload 2` ?

3 Information flow analysis

Question 5.[2pt]

Consider the following program:

$$\texttt{x} := \texttt{y};\ \texttt{if}\ (\texttt{x})\ \texttt{then}\ \{\texttt{y} := 1\}\ \texttt{else}\ \{\texttt{y} := 1\}$$

Each of the variables `x` and `y` is either high or low. This gives four possible combinations. For each of these four cases, explain if the program is non-interferent or not. Clearly justify your answers.

Question 6.[2pt]

Consider the following program:

$$\texttt{if}\ (\texttt{x})\ \texttt{then}\ \{\texttt{y} := 1\}\ \texttt{else}\ \{\texttt{x} := 1\}$$

In the case $(\texttt{x}, \texttt{y}) \in \mathbb{V}_L \times \mathbb{V}_H$, show that it is typable in the information flow type system defined in the course by giving a type derivation.

Question 7.[4pt]

The type system presented during the course is flow-insensitive: variables have the same security level at all program points.

Define a flow-sensitive information flow analysis. You may use any of the formalisms (data flow, constraint- or type-based, abstract interpretation) presented during the course.

The analysis should take indirect flows into account, and be able to prove the following program `l := h ; l := 0;` to be non-interferent.