Exercises

Program Analysis (CO70020)

Sheet 3

Exercise 1 *Consider the following* **while** *program:*

```
[y := 2]^1;
if [z > 1]^2
then [x := 1]^3
else [x := -1]^4;
[y := x * x]^5;
```

Perform a Constant Propagation Analysis CP *for this program.*

Exercise 2 *Consider the following simple imperative language with statements:*

$$S \quad ::= \quad \mathbf{skip} \;\Big|\; x := a \;\Big|\; S_1 \; ; \; S_2 \;\Big|\; \mathbf{if}\; b\; \mathbf{then}\; S_1 \;\mathbf{else}\; S_2 \;\Big|\; \mathbf{while}\; b \;\mathbf{do}\; S$$

Define an instance of the monotone framework (similar to the Constant Propagation Analysis CP*) which performs a usage analysis* Use *of expressions.*

> For each program point, the Use Analysis will determine the minimum and maximum number of program points the value of an arithmetic expression will be used when leaving this program point and before any variables in the expressions get redefined.

Assume that labelling, flow (flow) and reverse flow (flow^R), as well as initial and final labels are defined as usual. Let $\mathbf{AExp}_\star$ *be the set of arithmetic sub-expressions in a program S. The lattice of abstract states is given by:*

$$\widehat{\mathbf{State}} = (\mathbf{AExp}_\star \to \mathbf{N}_\infty \times \mathbf{N}_\infty)$$

with $\mathbf{N}_\infty = \{0, 1, 2, \ldots\} \cup \infty$*. These record (at every label) the number of times an expression might be used and the number of times it is guaranteed to be used. Define a least upper bound operator $\sqcup$ and $\sqsubseteq$ on $\widehat{\mathbf{State}}$ and identify the $\perp$ and $\top$ elements. Use $\widehat{\mathbf{State}}$ to define a* Use *Analysis like a monotone framework instance, i.e. specify the flow F, the extreme labels E, their initialization ι and the transfer functions. Derive the data-flow equations for the following program (after labelling it)*

```
y := a*b
while (x < a * b) do (
   x := a+b;
   y := a+b );
x := a*b;
```

Is this a may or a must analysis? How is it related to the VB *analysis? Discuss whether* $\widehat{\textbf{State}}$ *fulfills the Ascending/Descending Chain Conditions, and whether the* Use *Analysis is computable, or how it can be made computable.*

Exercise 3 *Consider the following program:*

$$[\texttt{x:=1}]^1; [\texttt{x:=x-1}]^2; [\texttt{x:=2}]^3$$

Clearly `x` *is dead at the exits from 2 and 3. But* `x` *is live at the exit of 1 even though its only use is to calculate a new value for a variable that turns out to be dead.*

> *We shall say that a variable is a* faint variable *if it is dead or if it is only used to calculate new values for faint variables; otherwise it is* strongly live*.*

In the example `x` *is faint at the exits from 1, 2 and 3.*
Define a Data Flow Analysis that detects Strongly Live variables.

Exercise 4 *Consider the following Fun program:*

$$\begin{array}{l}(\texttt{let}\ f = (\texttt{fn}\ z\ \texttt{=>}\ 1)\ \texttt{in}\\ \quad(((\texttt{fn}\ x\ \texttt{=>}\ x\ x)(\texttt{fn}\ y\ \texttt{=>}\ y))\ f))\end{array}$$

Label the program and give a brief and informal description of its execution: what's the result? Evaluate the expression formally using the eval function (from the lecture). For every step specify the environment ρ*.*