Analysis and Optimisation of Embedded Systems, SoSe 2015

Exercise 1

Due date: May 28, 2015

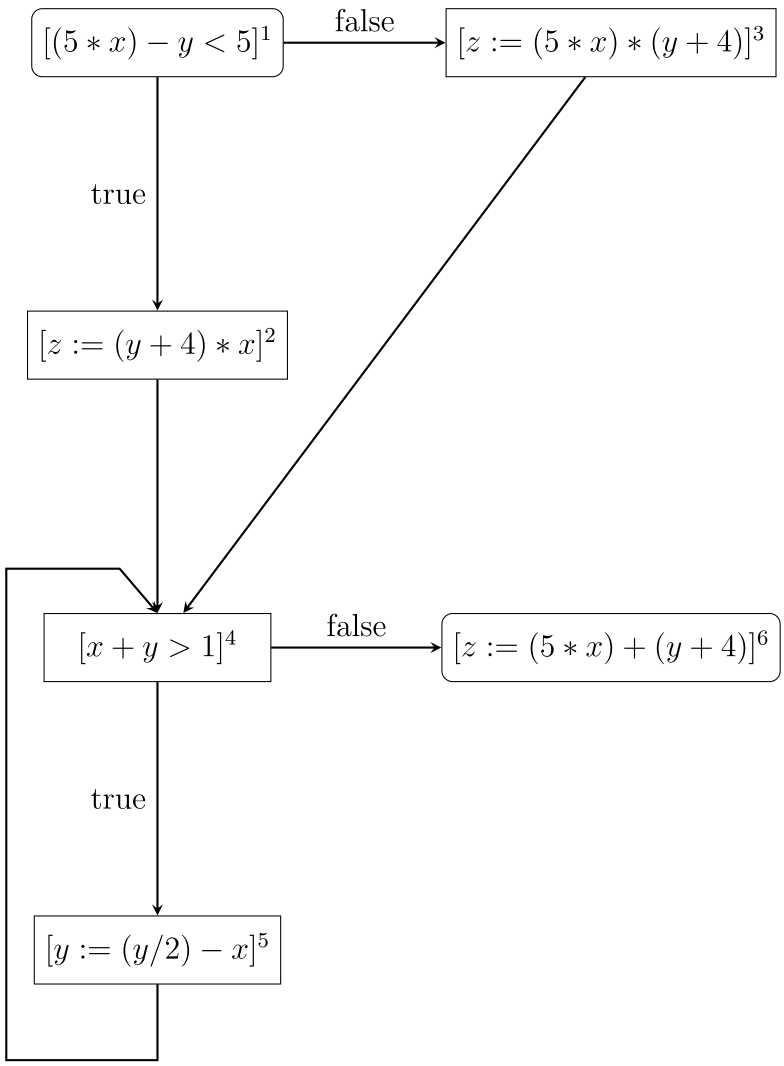
Submitted by

Rajibul Alam, r.alam@campus.tu-berlin.de
Tamilselvan Shanmugam, tamilselvan@mailbox.tu-berlin.de
Xu CAO, xu.cao@campus.tu-berlin.de

Last Modified: May 28, 2015

1 Available Expressions Analysis

1.1 Control flow graph



1.2 $\mathbf{AExp}_*$

Available expression for the given program is : **(5*x)** which is computed and not modified throught the program.

1.3 *kill* and *gen* table

l	$\text{kill}_{AE}(l)$	$\text{gen}_{AE}(l)$
1	$\emptyset$	{ (5*x), ((5*x)-y) }
2	$\emptyset$	{ (y+4), ((y+4)*x) }
3	$\emptyset$	{ (5*x), (y+4), ((5*x)*(y+4)) }
4	$\emptyset$	{ (x+y) }
5	{ (x+y), (y+4), ((5 * x)*(y+4)), ((y+4)*x), ((5*x)-y), (y/2), ((y/2)-x) }	$\emptyset$
6	$\emptyset$	{ (5*x), (y+4), ((5*x)+(y+4)) }

1.4 Equation system

AE$_\circ$(1) = $\emptyset$
AE$_\circ$(2) = AE$_\bullet$(1)
AE$_\circ$(3) = AE$_\bullet$(1)
AE$_\circ$(4) = AE$_\bullet$(2) $\cap$ AE$_\bullet$(3) $\cap$ AE$_\bullet$(5)
AE$_\circ$(5) = AE$_\bullet$(4)
AE$_\circ$(6) = AE$_\bullet$(4)

AE$_\bullet$(1) = AE$_\circ$(1) $\cup$ { (5*x), ((5*x)-y) }
AE$_\bullet$(2) = AE$_\circ$(2) $\cup$ { (y+4), ((y+4)*x) }
AE$_\bullet$(3) = AE$_\circ$(3) $\cup$ { (5*x), (y+4), ((5*x)*(y+4)) }
AE$_\bullet$(4) = AE$_\circ$(4) $\cup$ { (x+y) }
AE$_\bullet$(5) = AE$_\circ$(5) \(x+y), (y+4), ((5 * x)*(y+4)), ((y+4)*x),
((5*x)-y), (y/2), ((y/2)-x) $\cup$ $\emptyset$
AE$_\bullet$(6) = AE$_\circ$(6) $\cup$ { (5*x), (y+4), ((5*x)+(y+4)) }

1.5 Simplifying Equation

Let us consider

$a = 5 * x$
$b = (5 * x) - y$
$c = y + 4$
$d = (y + 4) * x$
$e = (5 * x) * (y + 4)$
$f = x + y$

Solution

$AE_\circ(1) = \emptyset$
$AE_\bullet(1) = \{a, b\}$

$AE_\circ(2) = \{a, b\}$
$AE_\bullet(2) = \{a, b, c, d\}$

$AE_\circ(3) = \{a, b\}$
$AE_\bullet(3) = \{a, b, c, e\}$

$$
\begin{aligned}
AE_\circ(4) &= \{a, b, c, d\} \cap \{a, b, c, e\} \cap AE_\bullet(5) \\
&= \{a, b, c\} \cap AE_\bullet(5) \\
AE_\bullet(4) &= \{a, b, c\} \cap AE_\bullet(5) \cup \{f\}
\end{aligned}
$$

$$
\begin{aligned}
AE_\circ(5) &= \{a, b, c\} \cap AE_\bullet(5) \cup \{f\} \\
AE_\bullet(5) &= (\{a, b, c\} \cap AE_\bullet(5) \cup \{f\}) \backslash \{b, c, d, e, f\} \\
&= \{a\} \cap AE_\bullet(5)
\end{aligned}
$$

Solutions for $\mathrm{AE}_\bullet(5)$ are **(5*x)** and $\emptyset$

1.6 Largest solution

l	$AE_\circ(l)$	$AE_\bullet(l)$
1	$\emptyset$	{ (5*x), ((5*x)-y) }
2	{ (5*x), ((5*x)-y) }	{ (5*x), ((5*x)-y), (y+4), ((y+4)*x) }
3	{ (5*x), ((5*x)-y) }	{ (5*x), ((5*x)-y), (y+4), ((5*x)*(y+4)) }
4	{ (5*x) }	{ (5*x), (x+y) }
5	{ (5*x), (x+y) }	{ (5*x) }
6	{ (5*x), (x+y) }	{ (5*x), (x+y), (y+4), ((5*x)+(y+4)) }

2 Live Variables Analysis

2.1 pseudo code of the worklist algorithm [1]

for all (v)
    $\mathrm{OUT}(v) = \emptyset$
    $\mathrm{IN}(v) = \mathrm{USE}(v)$
end for
worklist $\longleftarrow$ set of all nodes
while (worklist $\neq \emptyset$)
    pick and remove a node v from worklist
    $\mathrm{OUT}(v) = \bigcup_{s \in SUCC(v)} \mathrm{IN}(s)$
    oldin = $\mathrm{IN}(v)$
    $\mathrm{IN}(v) = \mathrm{USE}(v) \cup (\mathrm{OUT}(v) - \mathrm{DEF}(v))$
    if oldin $\neq \mathrm{IN}(v)$
        worklist $\longleftarrow$ worklist $\cup$ $\mathrm{PRED}(v)$
end while

2.2 Live variable analysis

Initializing the nodes

l	IN(l)	OUT(l)
1	{ x,y }	$\emptyset$
2	{ x,y }	$\emptyset$
3	{ x,y }	$\emptyset$
4	{ x,y }	$\emptyset$
5	{ x,y }	$\emptyset$
6	{ x,y }	$\emptyset$

W = 1,2,3,4,5,6   picked node = 1
W = 2,3,4,5,6

$$\begin{aligned}
OUT(1) &= \{x, y\} \cup \{x, y\} \\
&= \{x, y\} \\
oldIn &= \{x, y\} \\
IN(1) &= \{x, y\} \cup (\{x, y\} - \emptyset) \\
&= \{x, y\} \\
oldIn &= IN(1) \\
W &= \{2, 3, 4, 5, 6\}
\end{aligned}$$

picked node = 2
W = { 3,4,5,6 }

$$\begin{aligned}
OUT(2) &= \{x, y\} \\
oldIn &= \{x, y\} \\
IN(2) &= \{x, y\} \cup (\{x, y\} - \{z\}) \\
&= \{x, y\} \\
oldIn &= IN(2) \\
W &= \{3, 4, 5, 6\}
\end{aligned}$$

picked node = 3
W = { 4,5,6 }

$$
\begin{aligned}
OUT(3) &= \{x, y\} \\
oldIn &= \{x, y\} \\
IN(3) &= \{x, y\} \cup (\{x, y\} - \{z\}) \\
&= \{x, y\} \\
oldIn &= IN(3) \\
W &= \{4, 5, 6\}
\end{aligned}
$$

picked node = 4
W = { 5,6 }

$$
\begin{aligned}
OUT(4) &= IN(5) \cup IN(6) \\
&= \{x, y\} \\
oldIn &= \{x, y\} \\
IN(4) &= \{x, y\} \cup (\{x, y\} - \emptyset) \\
&= \{x, y\} \\
oldIn &= IN(4) \\
W &= \{5, 6\}
\end{aligned}
$$

picked node = 5
W = { 6 }

$$
\begin{aligned}
OUT(5) &= \{x, y\} \\
oldIn &= \{x, y\} \\
IN(5) &= \{x, y\} \cup (\{x, y\} - \{y\}) \\
&= \{x, y\} \\
oldIn &= IN(5) \\
W &= \{6\}
\end{aligned}
$$

worklist W = $\emptyset$
picked node = 6, W = { 5,4,3,2,1, }

$$\begin{aligned}
OUT(6) &= \emptyset \\
oldIn &= \{x, y\} \\
IN(6) &= \{x, y\} \cup (\emptyset - \{z\}) \\
&= \{x, y\} \\
oldIn &= IN(6) \\
W &= \emptyset
\end{aligned}$$

Live variables

l	IN(l)	OUT(l)
1	{ x,y }	{ x,y }
2	{ x,y }	{ x,y }
3	{ x,y }	{ x,y }
4	{ x,y }	{ x,y }
5	{ x,y }	{ x,y }
6	{ x,y }	$\emptyset$

2.3 Perform the program according to the small-step operational semantics

step0:

$$\frac{(5 * x - y < 5, [x = 4, y = 32, z = 3]) = tt}{\begin{array}{r}(if[5 * x - y < 5]then[z := (y + 4) * 4]else[z := (5 * x) * (y + 4)], [x = 4, y = 32, z = 3]) \\ \Rightarrow (z := (y + 4) * x, [x = 4, y = 32, z = 144])\end{array}}$$

step1:

$$\frac{(x + y > 1, [x = 4, y = 32, z = 144]) = tt}{\begin{array}{l}(while[x + y > 1]do[y := (y/2) - x], [x = 4, y = 32, z = 144]) \\ \quad \Rightarrow ([y := (y/2) - x], while[x + y > 1]do[y := (y/2) - x], [x = 4, y = 12, z = 144])\end{array}}$$

step2:

$$\frac{(x + y > 1, [x = 4, y = 12, z = 144]) = tt}{\begin{array}{l}(while[x + y > 1]do[y := (y/2) - x], [x = 4, y = 12, z = 144]) \\ \quad \Rightarrow ([y := (y/2) - x], while[x + y > 1]do[y := (y/2) - x], [x = 4, y = 2, z = 144])\end{array}}$$

step3:

$$\frac{(x + y > 1, [x = 4, y = 2, z = 144]) = tt}{\begin{array}{l}(while[x + y > 1]do[y := (y/2) - x], [x = 4, y = 2, z = 144]) \\ \quad \Rightarrow ([y := (y/2) - x], while[x + y > 1]do[y := (y/2) - x], [x = 4, y = -3, z = 144])\end{array}}$$

step4:

$$\frac{(x + y > 1, [x = 4, y = -3, z = 144]) = ff}{(while[x + y > 1]do[y := (y/2) - x], [x = 4, y = -3, z = 144]) \Rightarrow ([x = 4, y = -3, z = 144])}$$

step5:

$$(z := (5 * x) + (y + 4), [x = 4, y = -3, z = 144]) \Rightarrow ([x = 4, y = -3, z = 21])$$

3 Verification of Live Variables Analysis

3.1 Proof of Lemmas

3.1.1 if(S,s)⇒(S′,s′) then flow(S′) ⊆ flow(S)

case:i
$\langle S_1; S_2, s\rangle \rightarrow \langle S_1'; S_2, s\prime\rangle$ because $\langle S_1, s\rangle \rightarrow \langle S_1', s\prime\rangle$

$$\begin{aligned} flow(S_1, S_2) &= flow(S_1) \cup flow(S_2) \cup \{(l, init(S_2)) | l \in final(S_1)\} \\ &\supseteq flow(S_1') \cup flow(S_2) \cup \{(l, init(S_2)) | l \in final(S_1')\} \\ &= flow(S_1'; S_2) \end{aligned}$$

case:ii $\langle S_1; S_2, s\rangle \rightarrow \langle S_2, s\prime\rangle$ because $\langle S_1, s\rangle \rightarrow \langle S_2, s\prime\rangle$

$$\begin{aligned} flow(S_1, S_2) &= flow(S_1) \cup flow(S_2) \cup \{(l, init(S_2)) | l \in final(S_1)\} \\ &\supseteq flow(S_2) \end{aligned}$$

case:iii
⟨if $[\mathrm{b}]^l$ then S_1 else S_2, s⟩ → ⟨ S_1, s ⟩ because $B[\![b]\!] = true$

$$\begin{aligned} flow(if\,[b]^l\,then\,S_1\,else\,S_2,\ s) &= flow(S_1) \cup flow(S_2) \cup \{(l, init(S_1)), (l, init(S_2))\} \\ &\supseteq flow(S_1) \end{aligned}$$

case:iv
$\langle$if [b]^l then S$_1$ else S$_2$, s$\rangle \rightarrow \langle$ S$_2$, s $\rangle$ because $B[\![b]\!] = false$

$$\begin{aligned} flow(if\,[b]^l\,then\,S_1\,else\,S_2,\ s) &= flow(S_1) \cup flow(S_2) \cup \{(l, init(S_1)), (l, init(S_2))\} \\ &\supseteq flow(S_2) \end{aligned}$$

case:v
$\langle\, while\,[b]^l\,do\,S, s\rangle \rightarrow \langle S; while\,[b]^l\,do\,S, s\rangle$ because B$[\![b]\!]$ = true

$$\begin{aligned} flow(S; while\,[b]^l\,do\,S, s) &= flow(S) \cup flow(while\,[b]^l\,do\,S, s) \cup \{(l', l) | l' \in final(S)\} \\ &= flow(S) \cup flow(S) \cup \{(l, init(S))\} \cup \{(l', l) | l' \in final(S)\} \\ &\quad \cup \{(l', l) | l' \in final(S)\} \\ &= flow(S) \cup \{(l, init(S))\} \cup \{(l', l) | l' \in final(S)\} \\ &= flow(while\,[b]^l\,do\,S) \end{aligned}$$

3.1.2 if(S,s)⇒(S′,s′) then blocks(S′) ⊆ blocks(S)

case:1
$\langle S_1; S_2, s\rangle \rightarrow \langle S_1'; S_2, s\prime\rangle$ because $\langle S_1, s\rangle \rightarrow \langle S_1', s\prime\rangle$

$$\begin{aligned} blocks(S_1, S_2) &= blocks(S_1) \cup blocks(S_2) \\ &= blocks(S_1') \cup blocks(S_2) \\ &\supseteq blocks(S_1') \end{aligned}$$

case:ii $\langle S_1; S_2, s\rangle \rightarrow \langle S_2, s\prime\rangle$ because $\langle S_1, s\rangle \rightarrow \langle S_2, s\prime\rangle$

$$\begin{aligned} blocks(S_1, S_2) &= blocks(S_1) \cup blocks(S_2) \\ &\supseteq blocks(S_2) \end{aligned}$$

case:iii
$\langle$if $[\text{b}]^l$ then S_1 else S_2, s$\rangle \rightarrow \langle$ S_1, s $\rangle$ because $B[\![b]\!] = true$

$$\begin{aligned} blocks(if\ [b]^l\ then\ S_1\ else\ S_2,\ s) &= blocks(S_1) \cup blocks(S_2) \cup \{[b]^l\} \\ &\supseteq blocks(S_1) \end{aligned}$$

case:iv
$\langle$if $[\text{b}]^l$ then S_1 else S_2, s$\rangle \rightarrow \langle$ S_2, s $\rangle$ because $B[\![b]\!] = false$

$$\begin{aligned} blocks(if\ [b]^l\ then\ S_1\ else\ S_2,\ s) &= blocks(S_1) \cup blocks(S_2) \cup \{[b]^l\} \\ &\supseteq blocks(S_2) \end{aligned}$$

case:v
$\langle\, while\, [b]^l\, do\, S, s\rangle \rightarrow \langle S; while\, [b]^l\, do\, S, s\rangle$ because $\text{B}[\![b]\!] = \text{true}$

$$\begin{aligned} blocks(S; while\, [b]^l\, do\, S, s) &= blocks(S) \cup blocks(while\, [b]^l\, do\, S) \\ &= blocks(S) \cup blocks(S) \cup \{[b]^l\} \\ &= blocks(S) \cup \{[b]^l\} \\ &= blocks(while\, [b]^l\, do\, S) \end{aligned}$$

3.2 Proof of skip and seq composition

3.(b) case $([skip]^l, s_1) \Rightarrow s_1$

live $\models LV^{\subseteq}([skip]^l)$ and $s_1 \sim_{N(init([skip]^l))} s_2$

then, $N(init([skip]^l)) = N(l) = live_{entry}(l)$

$\supseteq live_{exit}(l) = X(l) = \emptyset$

thus, $s_1 \sim_{N(l)} s_2$

by taking $s_2' = s_2$ with $([skip]^l, s_2) \Rightarrow s_2'$ we get

$s_1' \sim_{X(l)} s_2'$

case $(S_1 ; S_2, s_1) \Rightarrow (S_2, s_1')$ with $(S_1, s_1) \Rightarrow s_1'$

since $(S_1, s_2) \Rightarrow s_2'$ with $s_1' \sim_{X(init(S_1))} s_2'$

and $(S_1, s_1) \Rightarrow s_1'$

then, $s_1' \sim_{N(int(S_2))} s_2'$ because

$X(init(S_1)) = N(init(S_2))$

thus, $s_1' \sim_{X(init(S_1))} s_2' = s_1' \sim_{N(init(S_2))} s_2'$

References

[1] Data Flow Analysis. http://www.cs.utexas.edu/users/mckinley/380C/lecs/05.pdf. Accessed: 2015-05-21.