Assignment 1 Solutions

Dataflow Analysis

Due: January 21, 11:00 am

This homework contains a combination of written assignment and gradiance quizzes. For the radiance quizzes please refer to http://www.gradiance.com. For the written assignment, every student must hand in his or her homework. Bring your homework to class on January 21. SCPD students may submit their homework by e-mail via scpd-distribution@lists.stanford.edu or give your homework to the courier.

1. [6 points] Is the following a meet operator? Please answer yes or no. No explanation is necessary.

   (a) [1 points] Maximum value (on integers)
   (b) [1 points] Product (on integers)
   (c) [1 points] Addition (on integers)
   (d) [1 points] Product mod 2 (on the set \{0, 1\})
   (e) [1 points] Addition mod 2 (on the set \{0, 1\})
   (f) [1 points] The GCD (Greatest Common Divisor) function (on integers)

Solution:

All the “no” answers failed because they were not idempotent (i.e., \(x \land x\) does not necessarily equal \(x\)).

   (a) Yes
   (b) No
   (c) No
   (d) Yes
   (e) No
   (f) Yes
2. [8 points] We say that a program point p belongs to a live range of a definition $d \ u=x+y$ iff the value assigned to u in definition $d$ may be accessed by using variable u at point p, for some path in the flow graph starting at p. Consider for example the code fragment below:

```
a = x + y;
b_0
  z = a;
b_1
  a = a + 1;
b_3
  b = a + z;
b_4
  a = 0;
b_2
```

The live range of the definition ‘$a = x + y$’ at point entrypoint($b_0$) is \{exitpoint($b_0$), entrypoint($b_1$), entrypoint($b_1$), exitpoint($b_3$), entrypoint($b_3$)}\}. Similarly, the live range of the definition ‘$a = a + 1$’ at point entrypoint($b_3$) is \{exitpoint($b_3$), entrypoint($b_4$)}\}.

This concept of live ranges can be used to increase flexibility in register assignment. For example, the live ranges of the definitions of $a$ in $b_0$ and $b_3$ do not intersect at any points, so it may be possible to use same registers to hold these two values.

Describe an algorithm to find the live range of every definition in a program.

**Solution:**

Compute both reaching definitions as well as liveness for each variable. A program point $P$ is in the live range of a definition $D : x = y + z$ if and only if $D$ reaches point $P$ and $x$ is live at the point $P$. 
3. [12 points] This question asks you to think about how changes to the initial values in a data flow analysis can affect the result. Recall that an answer to a data flow problem is considered ‘safe’ if it is no bigger than the ideal solution. Suppose you have defined a forward dataflow algorithm that is distributive and has finite descending chains. You accidentally initialized \( \text{OUT}[B] \) to \( \bot \) for all nodes other than \( \text{ENTRY} \).

(a) [3 points] Will your algorithm give a safe answer for all flow graphs?
(b) [3 points] If not, will it give a safe answer for some flow graphs? If it will, give an example.
(c) [3 points] Will your algorithm give the MOP solution for all flow graphs?
(d) [3 points] If not, will it give the MOP solution for some flow graphs? If it will, give an example.

Solution:

(a) Yes, the modified values start and remain \( \leq \) the MFP values.
(b) Yes, see previous.
(c) No.
(d) Yes, it will give the MOP on any acyclic flow graph.
4. [18 points] Consider a programming language that only has the following statements.

- Assignment: $x = a$
- Addition: $x = a + b$
- Output: output($x$)
- If-condition: if $(a > b)$ goto L

Your task is to eliminate assignments and additions that don’t contribute to the computation of output variables (DO NOT change control flow). For example, the following program

L1: i = 10
L2: j = 5
L3: k = 0
L4: i = i + 1
L5: j = j + 1
L6: k = k + 1
L7: if (k < 5) goto L4
L8: output(i)

should be simplified to the following after dead code elimination.

L1: i = 10
L2: k = 0
L3: i = i + 1
L4: k = k + 1
L5: if (k < 5) goto L3
L6: output(i)

Design a data flow analysis to implement this optimization.

(a) [2 points] What is the domain?
(b) [2 points] What is the meet operator?
(c) [2 points] What is the direction of data flow?
(d) [2 points] What is the initialization condition?
(e) [6 points] Describe the transfer function for a basic block.
(f) [4 points] Describe how you would eliminate dead code after running the data flow analysis.

Solution:

(a) Set of variables that contribute to the computation of output variables.
(b) Union.
(c) Backwards.
(d) $In[B] = \text{empty set.}$

(e) Assume a basic block contains only one statement. Let $f_s$ be the transfer function of statement $s$, and $m$ and $m'$ represent data-flow values such that $m' = f_s(m)$. We define $f_s$ based on the following different statements.

i. Assignment: $x = a$. If $x \in m$, $m' = m \setminus \{x\} \cup \{a\}$. Otherwise, $m' = m$.

ii. Addition: $x = a + b$. If $x \in m$, $m' = m \setminus \{x\} \cup \{a, b\}$. Otherwise, $m' = m$.

iii. Output: output($x$). $m' = m \cup \{x\}$.

iv. If-condition: if ($a > b$) goto L. $m' = m \cup \{a, b\}$.

(f) After running the data flow analysis, we can remove assignments and additions if the target variable is not in the variable set immediately after the statement.