CS 255/455 Homework 1
Spring 2014

February 19, 2014

Instructions:

- The homework is due 5pm Feb 26th in an envelope in Lingxiang Xiang's mailbox in CSB 7th floor mailroom and will be handed back in next Friday's help session (on Feb 28th). The mid-term exam will be handed out on March 3rd, with a similar format and content.
- **255 students who answer 455-only questions correctly will receive extra credits.**

Scores:

Question 1 :_____

Question 2 :_____

Question 3 :_____

Question 4 :_____

Total :_____

Question 1 Basic Concepts (20 points):

Give the definitions of the following concepts:

1. extended basic blocks
2. dominance frontier
3. data flow analysis
4. **(455 required, 255 extra credit)** monotonicity condition

Answer:

Question 2 Live Analysis (40 points):

The present GCC compiler can identify uninitialized variable uses when it is invoked with the optimization flag on. Given the example C program file, uninit.c,

```
int f(int b) {
  int i;
  return b==0?  i:  0;
}
```

When compiled by `gcc -c -O -Wuninitialized uninit.c`, the compiler generates
`uninit.c:3:  warning:  .i.  is used uninitialized in this function`

The warning is generated by data flow analysis, in particular, Live analysis. In fact, GCC cannot support the flag -Wuninitialized if -O is not specified.

1. Assuming you have the Live set of each basic block of a function, show how to identify the set of variables that may be used without initialization. You do not need to show how to compute the Live sets. To make the problem simpler, your augmented analysis is required to identify only the name of the variables, not the location of the uninitialized use.

2. **(455 required, 255 extra credit)** Data flow analysis conservatively assumes that all control flow paths through a control flow graph may happen, which leads to imprecision when some control flow paths are actually impossible due to program logic. Show that the -Wuninitialized feature in GCC may produce false positives by giving an example program in which no variable can be used before initialization but the compiler analysis still generates a warning of a possible use of uninitialized data.

Answer:

Answer Continued:

Question 3 SSA (40 points):

Give the control flow graph of the following pseudo code. List the basic steps of SSA (static-single assignment) form construction. Convert the code into the SSA form.

```
s = 1
r = 1
j = 2
while (j<n) {
   s = s + r
   j++
   if (j==n) {
     r = s
     break
   }
   r = s + r
   j++
}
print r
```

Answer:

Answer Continued:

Question 4 Immediate Dominator (20 points):

(extra credit for both 455/255) Prove that the immediate dominator (the first dominator in a control flow path from the entry block) is the same regardless of the actual execution path and therefore it is unique (except for the entry block, which has no immediate dominator).

Answer: