Homework 3: Intermediate representation

April 30, 2024

1.

```
int foo(int a) {
  int i;
  int d = 0;
  for (i = 0; i < a; i++) {
    if (i < 10) {
      d = foo(i) + bar(a);
    }
    else if (i < 20) {
      d = foo(i);
    }
    else {
      goto out;
    }
  }
  out:
  return d;
}
```

   Transform the above routine into an IR (use the IR that was discussed in class). Identify all basic blocks in the IR. Draw the CFG using the basic blocks in the IR. [10]

2. Extend the constant propagation algorithm discussed in the class to track all variables instead of a single variable. Clearly state the meet operator, initializer, and the transfer functions for each instruction in the IR (use the IR discussed in class). [10]

3. Given a basic-block B, write a static analysis to find all basic-blocks that are guaranteed to execute if B gets executed at runtime. You can assume that the program always terminates. [10]

4. Write a static analysis (at the function granularity) to identify all the instructions that are not needed to compute the return value of a function. For example, lines 5, 6, 7, 10, 11, and 12 are redundant in the following routine. (Use the IR discussed in the class.) [15]

```
1. int foo (int a, int b)
2. {
3.   int i, r1, r2, r3, r4;
4.   r1 = 0;
5.   r2 = 0;
6.   r3 = 0;
7.   r4 = 0;
8.   for (i = 0; i < 10000; i++) {
9.     r1 = a + b;
10.    r2 = i + a;
11.    r3 = r2 + b;
12.    r4 = r4 + 1;
13.  }
14.  return r1;
15.}
```

How to submit

Submit your handwritten answers in the class at the start of the lecture.