MC97 Homework 5

Due May 15, 1996; Max Hailperin

In this homework, you will carry out the process of register allocation by graph coloring, complete with coalescing, on a slight variant of the procedure considered in class. You will be using the iterated coalescing method described in the paper by George and Appel.

Here is the three-address intermediate code version of the procedure that you will be working on. We can assume that an invisible definition of the name `$a0` reaches the top of the program, and that the name `$v0` is live at the bottom of the program, because of an invisible use of `$v0` somewhere later on. The parenthesized numbers to the left of each assignment statement will be used to refer to them later in the homework.

```
fib:
(1)     n := $a0
(2)     old := 1
(3)     new := 1
(4)     i := 1

loop:
(5)     x1 := i >= n
        if x1 goto done

(6)     x2 := new
(7)     x3 := old + new
(8)     old := x2
(9)     new := x3
(10)    i := i + 1
        goto loop

done:
(11)    $v0 := new
```

1. Make a table with four columns and one row for each assignment in the three-address code. The first column should be the instruction number (the parenthesized number in front of the instruction). The second column should be the name defined by the instruction. The third column should be a list of the names used by the instruction. These first three columns are easy, so fill them in first.

   The fourth column should be a list of the names live "at the instruction" (see below). The most interesting point to use for liveness analysis is actually neither before nor after the instruction, but rather right in the middle, after the right hand side has been evaluated but before the assignment has taken place. This is the point you should use for the fourth column of your table. Should you start at the top of table and work your way down, or start at the bottom and work your way up? Whichever direction you work, don't forget that the code isn't all one long basic block—there is a loop. You should be able to do the necessary data flow analysis by informal understanding of the program and/or by analogy with the in-class example, though you could do it by real formal data flow analysis if you really wanted to.

2. Draw an interference graph with one node for each name in the three address code. You should base the edges of your graph on the table in problem 1. Label each edge with a list of instruction numbers, namely the ones that necessitate the edge. Two nodes are adjacent if one of them is live at a definition point of the other. However, you should not make the source and target of a copy instruction interfere with each other based on that instruction, even if the source continues to be live. You should add an additional edge between the `$v0` and `$a0` nodes, since the intent is that these names be precolored with the corresponding (different) registers. Once these edges are in place, add a dotted edge between each pair of move-related nodes that do not have a (solid) interference edge already between them. Label the dotted edges with the instruction number(s) of the move(s) done between that pair of names. For your own use in the following part of the homework, you'll also want a pencil copy of this graph from which you can progressively erase.

3. Color the coalesced graph using five colors, namely `$a0`, `$v0`, `$t0`, `$t1`, and `$t2`, using the procedure from the George and Appel paper. No freezing or spilling is necessary; only simplify and coalesce steps, followed by select steps to assign the actual (non-precolored) colors.

   Show your work using a two column table. The first row will be for the first simplify or coalesce step you do, the second row will be for the second simplify or coalesce step, etc. You will fill the first column of the table in, working down the table, and then turn around and fill the second column in headed back up the table.

   As you work down the table, fill in the first column with the step you have chosen to do, for example, "simplify A" or "coalesce A and B." For each row, you may freely choose any legal simplify step, and if there is none, freely choose any legal coalesce step. As you go, use your pencil (particularly the eraser end) to update your penciled copy of the graph. You should be able to reduce the whole graph to nothing but precolored nodes in this way.

   Now turn around and head back up the table from bottom to top, doing the select phase of the algorithm. For each row that is of the form "simplify A," add A back into your penciled graph and connect it back up with those of its neighbors that are currently in the penciled graph. Choose a color (register) for the name (or coalesced name group) A. You may choose any of the five registers that doesn't doesn't conflict with the coloring of A's neighbors that are currently in the penciled graph. Put the register (color) you chose in the second column of your table. Skip over the rows that are of the form "coalesce A and B."