CS 302 : Implementation of Programming Languages
TUTORIAL 9 (Optimization) April 05, 2017

P1. Construct a control flow graph for the following intermediate code

```
10 :  t1 = x + z
11 :  t2 = y + u
12 :  if t1 > t2 goto  14
13 : goto  21
14 : if  x > z  goto  16
15 : goto  27
16 : t3 = x  + y
17 : x = t3
18 : t4 = y / 2
19 : y = t4
20 : goto 14
21 : t5 = x * y
22 : x = t5
23 : t6 = 2 * y
24 : y = t6
25 : if y > x  goto 21
26 : goto  27
27 : t7 = x / y
28 : x = t7
29 : if x > 0  goto 10
30 : goto 31
31 : return
```

P2. Consider the quicksort() function in C agiven below. Assume SDTS used in the course for the linguistic features present in the function and feel free to extend the SDTS as required.

```
void quicksort(int m, int n)
{ int i, j, v, x;
if (n <= m) return;
i = m  1; j = n; v = a[n]; /* v is the pivot */
while(1) /* Move values smaller */
{ do i = i + 1; while (a[i] < v); /* than v to the left of */
   do j = j  1; while (a[j] > v); /* the split point (sp) */
   if (i >= j) break; /* and other values */
   x = a[i]; a[i] = a[j]; a[j] = x; /* to the right of sp */
} /* of the split point */
        x = a[i]; a[i] = a[n]; a[n] = x; /* Move the pivot to sp */
quicksort(m,i); quicksort(i+1,n); /* sort the partitions to */
} /* the left of sp and to the right of sp independently */
```

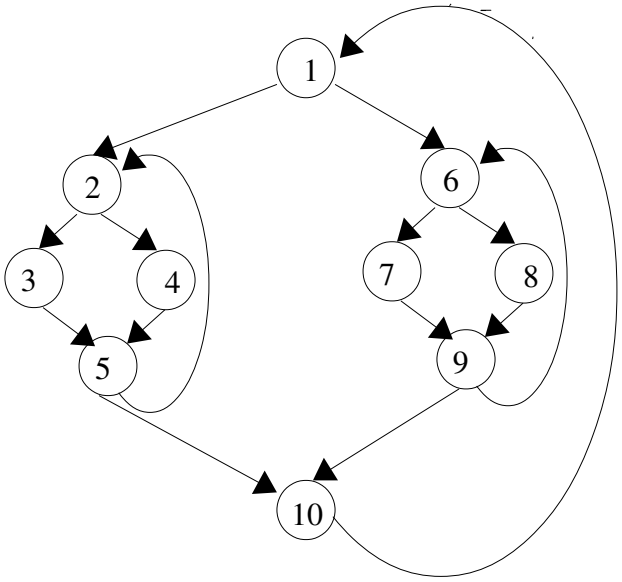
(a) Generate the intermediate code sequence produced by your SDTS after semantic analysis of the code given.

(b) Manually process the intermediate code of part(a) above to i) Identify the headers of basic blocks, ii) construct the basic blocks for each header and iii) show the control flow graph (cfg)

(c) Augment the SDTS used for part (a) above so that the list of headers are generated along with parsing and semantic analysis. Explain the attributes used by you for identifying the leaders and how backpatching is done if relevant.

(d) Use the SDTS of part (a) to generate the headers and report.

P3. Find the dominators of each node in the following cfg. Use the recursive equations for DOM(i) and the iterative algorithm for solving it. The iterative algorithm should process the nodes in reverse DFS order. Also draw the dominator tree from the information computed.



P4. It is intended to find the set of nodes dominated by a node i, to be denoted by DOMINATED_BY(i) for all nodes i of a cfg. You have to formulate recursive equations along with the initializations so that solution of these equations directly generate the information sought. Justify the equations and the initializations. Use your formulation to compute the DOMINATED_BY(i) information for the diamond graph of problem P3.

P5. Consider the following program

```
int a, b, c, d;
p()
{
        a = b + c;
        while (b < c)
        { while (a > c)
                {
                        b = c + d;
                        c = d + b;
                }
          a = b + d;
        }
        return a;
}
```

(a) Construct a control flow graph in terms of basic blocks

(b) Perform live variables analysis on the control flow graph. Show the trace of your analysis in the form of the following table:

Node	Succ	GEN	KILL	Initial Value	Iteration 1	…..	Iteration n
1							
...							
..							

P6. Determine the Reaching Definitions data flow information for the program fragment given in P5.

P7. Determine the Available Expressions information for the program fragment given in P5.

P8. Assume that the information about live variables, reaching definitions and available expressions are known at the entry of each basic block. Give an outline of the optimizing transformations that can be performed on node i and also point out the kind of improvements that are possible.

Supratim Biswas, Posted on April 5, 2017