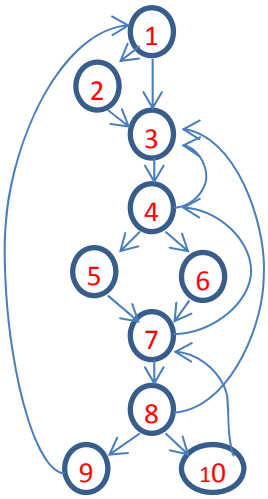
Examples of Control Flow Analysis

1. DFST and node numbering

We start by finding a depth-first-spanning tree (DFST) for a rooted directed graph, such as the one shown in Figure 9.38. In fact, for the purpose of program analysis, we normally add a unique entry node that has no incoming edges and a unique exit node that has no outgoing edges. However, this example serves the purpose of illustrating node numbering and DFST all right.

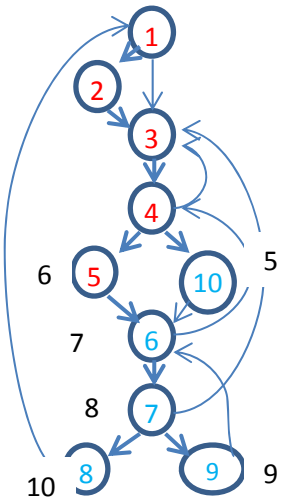


Given the flow graph to the left, we obtain a DFST by performing a depth first search (DFS) starting at the root (node 1). The edges through which the search goes forward are called the tree edges that form the spanning tree. The tree edges are drawn thinker in the graph below. The nodes are renumbered by the preorder number based on this DFST. The changed numbers are colored in blue.

Based on this DFST, we have one forwarding edge (also known as advancing edge in the textbook) that is (1, 3). We have retreating edges (8,1), (9,6), (6,4), (4,3) and (7,3). We have a cross edge (10,6).

We can also use this example to illustrate the reversed post-order numbering, which assigns a number to a node immediately before returning to its parent in the DFST during the DFS. The number is assigned in the decreasing order. The rpost numbers, if different from the preorder number, are marked next to the nodes in black.

We can see that, except for the retreating edges, every edge is from a node with a lower rpost number to one with a higher rpost number. If we remove the retreating edges, we obtain a directed acyclic graph (DAG) that covers the original graph. We can see that in this DAG, the rpost numbers form a top sort.



2. Dominators

We now use a work list to compute the dominance relationship on the flow graph listed above. Initially all nodes are placed in the work list. For simplicity we will place them in the order of the rpost numbering. The set of dominators is initialized as containing all the nodes except for the node 1 (*we use the rpost numbers here*) whose dominator is just node 1. The following table shows the computation results after applying the formula (i.e. the transfer function) "dom(n) = {n} union (intersection of dom(p) for all predecessors, p, of n) " to every node n in the work list.6

n	2	3	4	5	6	7	8	9	10
Dom(n)	1,2	1,3	1,3,4	1,3,4,5	1,3,4,6	1,3,4,7	1,3,4,7,8	1,3,4,7,8,9	1,3,4,7,8,10
Add to wlist			3			4		7	1

n	3	4	7	1
Dom(n)	1,3	1,3,4	1,3,4,7	1

From the table, we see that the dominators always have lower rpost numbers than n has (except for n itself of course). As a matter of fact, we know the immediate dominator is of the highest rpost number among the dominators (except n itself). We omit the construction of the dom tree here because it is simply connecting each node upward to its immediate dominator.

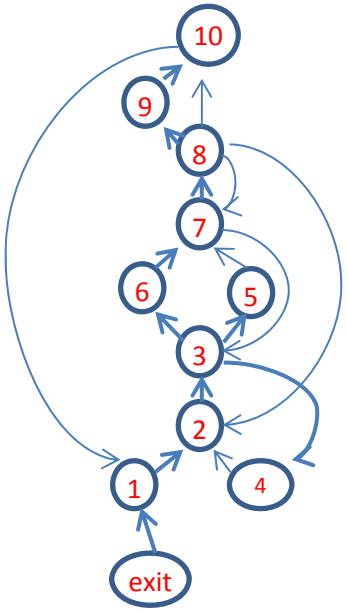
3. Post dominator

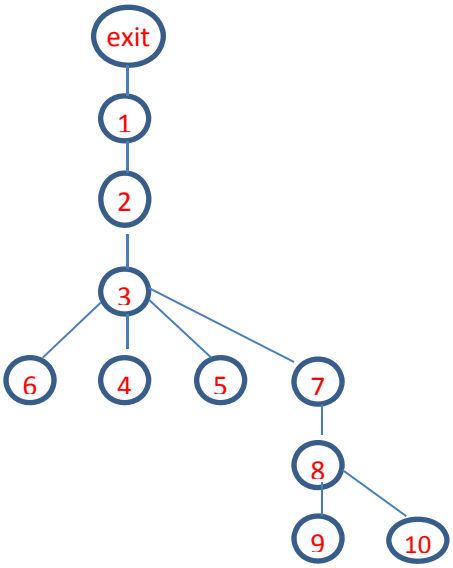
To use the same flow graph (as above) for illustrating the construction of the post dominator tree, we need to insert an exit node to the graph. We arbitrary assume that the node with rpost number 10 has an outgoing edge to the exit node. We reverse the result flow graph, as shown on the next page. The solid edges form a DFST for this reversed flow graph. The red numbers are rpost numbers according to this DFST. The exit node has the rpost number 0.

We use a work list that initially includes all the nodes. For convenience, we place the nodes according to the newly computed rpost numbers. Every node except the exit node has dominators (i.e. the post dominators in the original graph) that include all nodes. The exit node is dominated by itself.

n	1	2	3	4	5	6	7	8	9	10
Dom(n)	Exit,1	Exit,1,2	Exit,1,2,3	Exit,1,2,3,4	Exit,1,2,3,5	Exit,1,2,3,6	Exit,1,2,3,7	Exit,1,2,3,7,8	Exit,1,2,3,7,8,9	Exit,1,2,3,7,8
Add to wlist				2			3	7		1

n	2	3	7	1
Dom(n)	Exit,1,2	Exit,1,2,3	Exit,1,2,3,7	Exit,1





The dominator tree for the reversed flow graph is shown to the right in the above. If we remap the nodes to the original numbering in Figure 9.38 in the textbook, the tree we have just built is the pdom tree of the given flow graph.

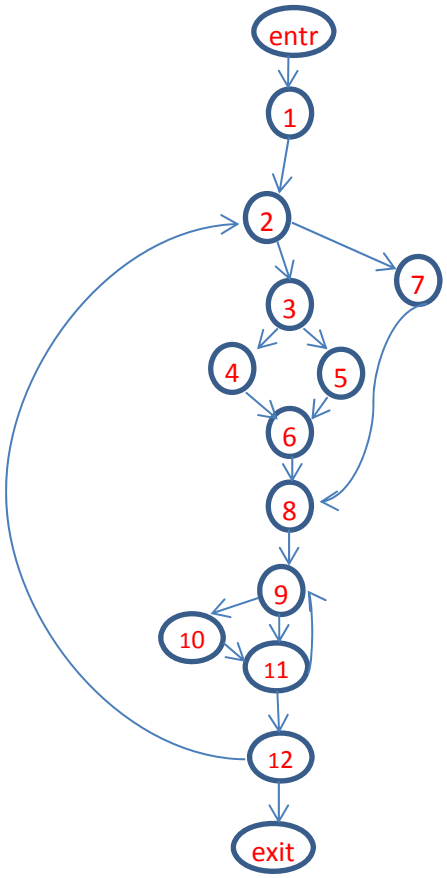
4. The dominance frontier and post-dominance frontier

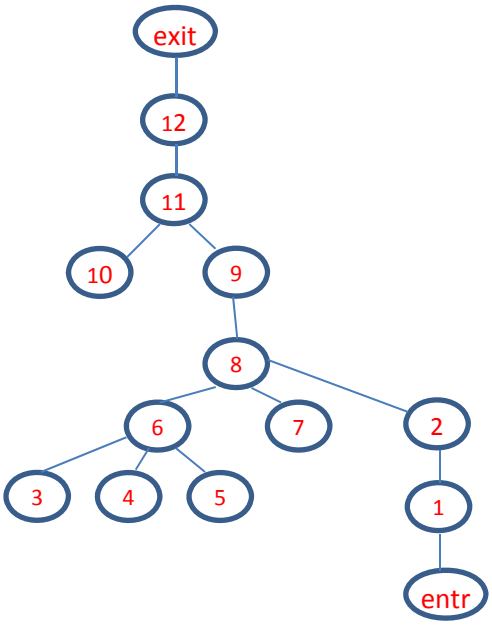
We compute the dominance frontiers (DF) for the nodes in the reversed flow graph (top left on this page), they will be the post-dominance frontiers for the original graph. To compute DF for the reversed flow graph, we talk through the dom tree shown on the top right on this page, using its numbering to identify the nodes. (The dominance frontier of exit in the reversed graph is empty.)

n	6	4	5	9	10	8	7	3	2	1
DF_local	7	2	7	10	1	7,2	3			
DF_up						1 (from 10)	1,7 (from 8)	2(from 4), 1,3(from 7)	1,2(from 3)	1(from 2)
DF(n)	7	2	7	10	1	1,2,7	1,2,3,7	1,2,3	1,2	1

If we remap the nodes to those in the original graph, DF(n) computed above gives us PDF(n) in the original graph. PDF(n) contains all the nodes on which n has a control dependence. This can be seen to be true in the flow graph according to the definition of control dependences.

The following is another example. This time we directly draw its PDOM tree (to the right) and computes the PDF.





n	10	3	4	5	6	7	entry	1	2	8	9	11	12	exit
PDF_local	9	2	3	3		2			12		11			
PDF_up					2 (from 3)					12(from 2)	12	11,12(from 9)	12	
PDF(n)	9	2	3	3	2	2			12	12	11,12			