

Finding Cut Vertices and Biconnected Components with Depth First Search

This algorithm uses depth first search to traverse a simple graph and detect which vertices are cut vertices (articulation points).

```

begin_DFS_main
  S ← ∅ ; count ← 0
  for each vertex u ∈ V(G)
    color[u] ← white ; Append(u, S)
  endfor
  while (S ≠ ∅)
    w ← Front(S) ; p[w] ← nil ; DFS(w)
  endwhile
end_DFS_main

begin_DFS(u)
  color[u] ← gray ; Delete(u, S)
  count ← count + 1 ; n[u] ← count
  low[u] ← count
  for each vertex v adjacent to u
    If color[v] = white
      then p[v] ← u ; DFS(v)
      low[u] ← min(low[u], low[v])
      If low[v] ≥ n[u]
        then print "u is an articulation point"
    else if v ≠ p[u]
      then low[u] ← min(low[u], n[v])
  endfor
  color[u] ← black
  return
end_DFS()

```

This algorithm uses depth first search to traverse a simple graph and partitions its edges into biconnected components. The components are organized as lists $BC[1], \dots, BC[num_comp]$ where num_comp is the number of biconnected components and $BC[i]$ is the list of edges in the i^{th} component. Additions to the standard DFS are in boxes.

```

begin_DFS_main
   $S \leftarrow \emptyset$  ; count  $\leftarrow 0$  ;  $num\_comp \leftarrow 0$  ;  $K \leftarrow \emptyset$  /*  $K$  is a stack */
  for each vertex  $u \in V(G)$ 
    color[ $u$ ]  $\leftarrow white$  ; Append( $u, S$ )
  endfor
  while ( $S \neq \emptyset$ )
     $w \leftarrow Front(S)$  ;  $p[w] \leftarrow nil$  ; DFS( $w$ )
  endwhile
end_DFS_main

begin_DFS( $u$ )
  color[ $u$ ]  $\leftarrow gray$  ; Delete( $u, S$ )
  count  $\leftarrow count + 1$  ;  $n[u] \leftarrow count$ 
   $low[u] \leftarrow count$ 
  for each vertex  $v$  adjacent to  $u$ 
    If color[ $v$ ] = white
      then  $push(uv, K)$ 
         $p[v] \leftarrow u$  ; DFS( $v$ )
        
 $low[u] \leftarrow \min(low[u], low[v])$ 
          If  $low[v] \geq n[u]$ 
            then  $num\_comp \leftarrow num\_comp + 1$  ;  $BC[num\_comp] \leftarrow \emptyset$ 
               $e \leftarrow pop(K)$  ; Append( $e, BC[num\_comp]$ )
              while( $e \neq uv$ )
                 $e \leftarrow pop(K)$  ; Append( $e, BC[num\_comp]$ )
              endwhile
            else if  $v \neq p[u]$ 
              then  $low[u] \leftarrow \min(low[u], n[v])$ 
                if  $n[u] > n[v]$  then  $push(uv, K)$ 
          endif
        
      endif
    endif
  endfor
  color[ $u$ ]  $\leftarrow black$ 
  return
end_DFS()

```