

Data: Function f

Result: Add "free" statement to release block-scope variables $block_vars$

```
1 Let INPUTS be the input parameters of function  $f$ ;  
2 Let CFs be a list of code fragments from function  $f$ ;  
3 Let BTree  $\leftarrow$  buildBlockTree( $CFs$ ) be a block-level tree, where  $CFs$   
   are scoped into a block, and blocks are clustered into a tree structure;  
4 computeBlockVars ( $BTree.root$ );  
1 Procedure computeBlockVars( $b$ )  
2    $outside\_vars[b] := \mathbf{INPUTS} \cup_{p \text{ is a parent of } b} block\_vars[p]$  ;  
3    $block\_vars[b] := outside\_vars[b]$ ;  
4   foreach  $cf$  in  $b$  other than ending block do  
5      $in[cf] := \cup_{cf_p \text{ is a predecessor of } cf} out[cf_p]$ ;  
6      $out[cf] := gen[cf] \cup (in[cf] - kill[cf] - transfer[cf])$ ;  
7     foreach  $var$  in  $kill[cf]$  do  
8       if  $var \in in[cf]$  then  
9         freeVar( $var$ );  
10      end  
11    end  
12  end  
13   $block\_vars[b] := in[end] - outside\_vars[b]$ ;  
14  foreach  $var$  in  $block\_vars[b]$  do  
15    freeVar( $var$ );  
16  end  
17  foreach child block  $cb$  in  $b$  do  
18    computeBlockVars( $cb$ );  
19  end
```