```
delete pseudocode
set CLEANUP and isSimpleDelete flags to false
while(true)
{
   do primarySeekForDelete(deleteKey)
   if(!CLEANUP)
   {
      try CAS(node->lChild,<nlChildAddr,0,0>,<nlChildAddr,1,0>)
      if CAS FAILED
         help
         continue from top of primary seek's while loop
      if CAS SUCCEEDED
         set CLEANUP to true and set storedNode = node
   }
   if(storedNode != node) //Someone removed the node for me. So DONE
   set "deleteFlag" on node's rChild using BTS
   if complex delete
   {
      while(true) //secondary seek
      {
>>       nRChild = node->rChild
>>       if(nRChild != NULL)
>>          isSplCase = secondarySeekForDelete(nRChild)
>>       else
>>          set isSimpleDelete flag to true
>>          break from while loop
         if node key is unmarked
         {
            try CAS(rnode->lChild,<NULL,0,0>,<nodeAddr,0,1>)
            if CAS failed
            {
               if promoteFlag is set
                  if address does not match with node's address
                     restart primary seek. assert(node->secFlag == DONE)
                     break from while loop
               else
                  if address != NULL //restart secondary seek
>>                continue from top of secondary seek's while loop
                  else
                     assert(rnode->lChild's deleteFlag is set)
                     help operation at secondaryLastUnmarkedEdge
                     if secondaryLastUnmarkedEdge does not exist, then help node->rChild
>>                continue from top of secondary seek's while loop
            }
            set promote flag on rnode->rChild using BTS
            promote key using a simple write. Node's key changes from <0,kN> to <1,kRN>
         }
         if(!isSplCase)
         {
            try CAS(rpnodeLChild,<rnode,0,0>,<rnodeRChild,0,0>) //remove secondary node
            if CAS FAILED, help operation at secondaryLastUnmarkedEdge
               if secondaryLastUnmarkedEdge doesn't exist, override CASinvariant and help
               node->rChild
>>          continue from top of secondary seek's while loop
            if CAS SUCCEEDED, set node->secDoneFlag to true
         }
         else
         {
>>          try CAS(nodeRChild,<rnode,1,0>,<rnodeRChild,1,0> //no problem if CAS fails
>>          set node->secDoneFlag to true
         }
>>       if(node->secDoneFlag is set)
         {
            create a fresh copy of node
            newNodeKey as <0,kRN>
            newNodeLChild as <node's lChildAddr,0,0>
            newNodeRChild = <node's rChildAddr,0,0>
            try CAS(pnode->lChild,<node,0,0>,<newNode,0,0>)
            if CAS SUCCEEDED then DONE
            if CAS FAILED
               if address has changed,then someone helped me install a fresh copy.so DONE
               else CAS has failed coz the edge is marked. Help at lastUnmarkedEdge.
                  if lastUnmarkedEdge is (pnode,node) then restart
                  break from while loop //start from primary seek
         }
      }
   }
   else //simple delete
      set isSimpleDelete to true
   if(isSimpleDelete)
   {
      try CAS(pnode->lChild,<node,0,0>,<node's l/r child,0,0>)
      if CAS SUCCEEDED, then DONE
      if CAS FAILED
         if lastUnmarkedEdge is NOT (pnode,node) help
   }
}
```