```
delete pseudocode
set CLEANUP, RESTART and isSimpleDelete flags to false
while(true)
{
  set RESTART to false
  do primarySeekForDelete(deleteKey)
  if(!RESTART)
  {
    if(!CLEANUP)
      try CAS(node->lChild,<nlChildAddr,0,0>,<nlChildAddr,1,0>)
      if CAS FAILED
        help and then set RESTART to true
      if CAS SUCCEEDED
        set CLEANUP to true and set storedNode = node
    if(!RESTART)
    {
      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
        {
          assumeCASsucceeded = false;
          SECONDARY_RESTART = false;
          isSplCase = secondarySeekForDelete(node);
          if node key is unmarked
          {
            try CAS(rnode->lChild,<NULL,0,0>,<nodeAddr,0,1>)
            if CAS failed
              if promoteFlag is set
                if address matches with node's address
                  set assumeCASsucceeded to true
                else
                  restart primary seek. assert(node->secFlag == DONE)
                  break from while loop
              else
                if address != NULL //restart secondary seek
                  set SECONDARY_RESTART to true;
                else
                  assert(rnode->lChild's deleteFlag is set)
                  help operation at secondaryLastUnmarkedEdge
                  if secondaryLastUnmarkedEdge does not exist, then help node->rChild
                  break from while loop //start from primary seek
            else //CAS succeeded
              set assumeCASsucceeded to true
            if(assumeCASsucceeded)
              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(!SECONDARY_RESTART)
          {
            if(!isSplCase)
              try CAS(rpnodeLChild,<rnode,0,0>,<rnodeRChild,0,0>) //remove secondary node
              if CAS FAILED, help operation at secondaryLastUnmarkedEdge
                if secondaryLastUnmarkedEdge does not exist, then override CASinvariant and help
```

```
                node->rChild
                break from while loop //start from primary seek
              if CAS SUCCEEDED, set node->secDoneFlag to true
            else
              set node->secDoneFlag to true
            if(node->rChild != NULL OR node->secDoneFlag is set)
            {
              create a fresh copy of node
              newNodeKey as <0,kRN> ... newNodeLChild as <node's lChildAddr,0,0>
              if(isSplCase)
                newNodeRChild as <rnodeRChild,0,0>
              else
                newNodeRChild = <nodeRChild,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 //complex delete changed to a simple delete
            {
              isSimpleDelete = true;
              break from while loop //This will go into simple delete
            }
          }
        }
      }
      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
      }
    }
  }
}
```