Assignment 3

Report

Dominique Roduit

1 – Requirements of a thread-safe linked list

To be thread-safe, multiple threads should be able to perform any operations on the linked list in parallel without affecting functionality and correctness of the list's operations. As we saw in the lecture, we could simply lock the whole list for every operation. But we would have very bad performances. In the next section I explain how I proceed to have fine grained locking.

2 – Describe how your additions apply to each operation

First, I added a lock in the node structure. Because every node should be able to be locked.

Then, everytime I add a new node to the list (init_list, append, add_first, insert), I init the lock of the new added node. In the same way, I destroy the lock for every deleted node (pop, remove_by_index, remove_by_value). Furthermore, for every insertion, I lock the nodes one after the others until I reach the desired position to add the new node, and for every deletion, I lock the node to remove and the previous one.

Finally, in a nutshell, if I had to sum up the main rules to follow in order to implement a thread-safe single linked list, I would say :

- When traversing the list (adding a new node, printing the list, counting)
  Lock the nodes one after the others.
  Lock « current », and release « previous » as and when.
  In the loops : previous = current ; current = current→next

- When removing a node
  Lock the node you want to remove and the previous one.

To be careful :

- Always make sure that « node » is locked when accessing to « node→next ». We don't need to lock node to access « node→val » for instance, because the val will never be changed by another thread, only « node→next » might be changed by another thread.

- Lock next node before to unlock current node

▪ pop

We don't need to lock the node after the head. Only the head can be locked, because if a thread want to remove the next node, then it will lock also the previous node, that is the head.

▪ delete_list

When deleting the list, we first lock all the nodes of the list before deleting them.