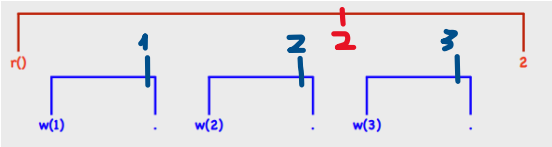
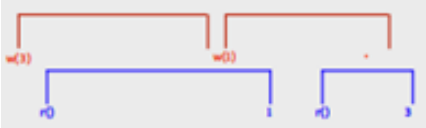
Concurrent Algorithms and Data Structures – Theory Assignment 1

Bernat Xandri Zaragoza

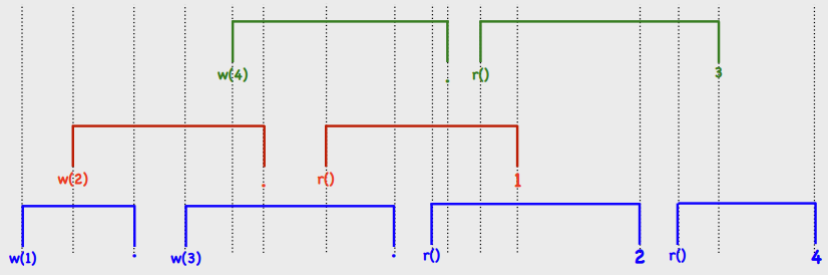
1. Problem 1:



In the first image, we see the linearizable history.



In this second image, we can see that it is not a linearizable history given the facts that linearizability requires that a read operation returns either the value of the last completed write operation. Therefore, since the first read operation (blue line) returns 1, which is incorrect given the fact that this value has not been written yet, we can say that this Register ADT is not linearizable.

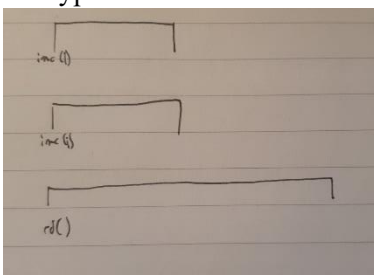


In this image, we can see that this is also not linearizable. This is due to the first read returning 1, when this value has been overwritten already.

2. Problem 2:

a) This abstract data type can be given as a set of N registers called R_i that are Register ADT, with 2 operations, which are the already described inc() and rd(). We must assume that, as it was the case in the first exercise, all Registers are initialized with value 0.

b) In this case, if we have 2 threads doing a inc operation concurrently, while a third thread does the rd operation, if rd reads the first's thread register before it has been updated and the second after it has been updated, we would observe a library history that is not linearizable wrt. the counter data type.



c) If each Ri is atomic, which implies that the calls to Ri cannot overlap, then the counter will be linearizable wrt. to the counter data type. The linearization policy will be that inc can be linearized at the point where Ri has been written, and rd only at the end of the execution, when the registers have all been read. This is because, when dealing with atomic operations, it ensures that no two operations on the same register can occur at the same time.