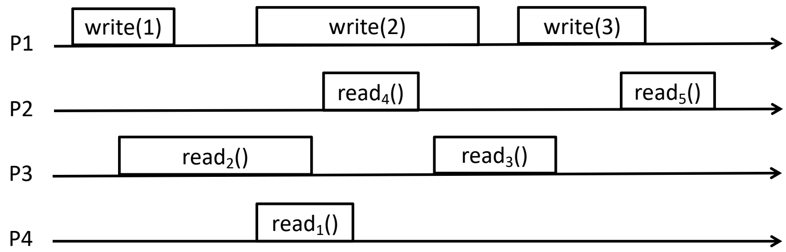
Dependable Distributed Systems
Master of Science in Engineering in Computer Science

AA 2023/2024

Week 9 – Exercises
November 22nd, 2023

Ex 1: Consider the execution depicted in the following figure and answer the questions



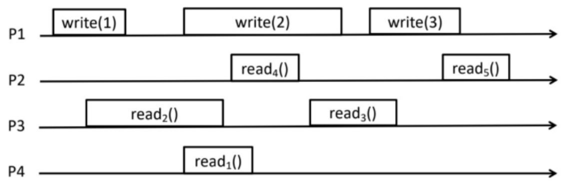
1. Define <u>ALL</u> the values that can be returned by read operations (Rx) assuming the run refers to a regular register.
2. Define <u>ALL</u> the values that can be returned by read operations (Rx) assuming the run refers to an atomic register.
3. Let us assume that values retuned by read operations are as follow: $read_1() \rightarrow 2$, $read_2() \rightarrow 2$, $read_3() \rightarrow 3$, $read_4() \rightarrow 1$, $read_5() \rightarrow 3$. Is the run depicted in the Figure linearizable?

Ex 2: Consider a distributed system constituted by *n* processes $\prod=\{p_1, p_2\ldots p_N\}$ with unique identifiers that exchange messages through perfect point-to-point links and are structured in a ring topology (i.e., each process p_i can exchange messages only with processes $p_{(i+1)\mathrm{mod}\ N}$ and stores its identifier in a local variable next).
Each process p_i knows the initial number of processes in the system (i.e., every process p_i knows the value of *N*).

1. Assuming that processes are not going to fail, write the pseudo-code of an algorithm that implements a (1, N) regular register.

Ex.3: Describe what an overlay network is and describe the two main classes that exist, highlighting their specific advantages and disadvantages.

Ex 1: Consider the execution depicted in the following figure and answer the questions



1. Define ALL the values that can be returned by read operations (Rx) assuming the run refers to a regular register.
2. Define ALL the values that can be returned by read operations (Rx) assuming the run refers to an atomic register.
3. Let us assume that values retuned by read operations are as follow: $read_1() \rightarrow 2$, $read_2() \rightarrow 2$, $read_3() \rightarrow 3$, $read_4() \rightarrow 1$, $read_5() \rightarrow 3$. Is the run depicted in the Figure linearizable?

1) Regular Register → R_1 : 1,2
R_2 : 0,1,2
R_3 : 1,2,3
R_4 : 1,2
R_5 : 2,3

2) Atomic Register → R_1 : 1,2
R_2 : 0,1,2
R_3 : 1,2,3 o 2,3
- 2,3 if $R_2=(2)$ $R_4=(2)$ $R_1=(2)$
- else 1,2,3

R_4 : 1,2 o 2
- 2 if $R_2=(2)$
- else 1,2

R_5 : 2,3 o 3
- else 2,3
- 3 if $R_3=(3)$

3) $R_1=2$ $R_2=2$ $R_3=3$ $R_4=1$ $R_5=3$

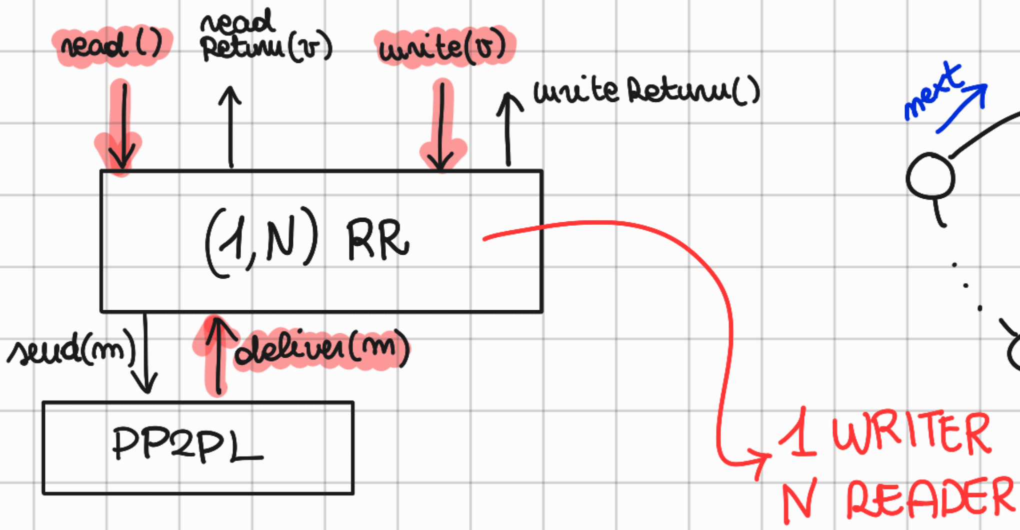
No, it is not linearizable because R_4 returns 1 after $R_2=2$

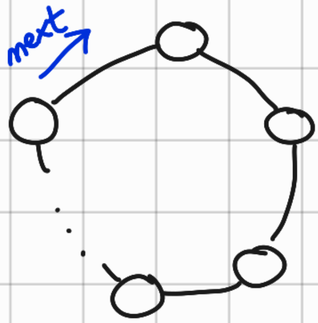
Ex.3: Describe what an overlay network is and describe the two main classes that exist, highlighting their specific advantages and disadvantages.

An overlay network is a network build on existing phisical network. The links are said to be logical or virtual, as they are usually indipendent of the underlying network links and topology. This means that 2 direct neighbors in the overlay, may be separated by several hops in the underlay. There are several classes: structured, means that nodes are organized in a specific structure like a ring; the schema is deterministic, so the position of the nodes is known a priori. While the unstructured has a random topology and the schema is impossible to predict.
In the first class we have an efficient routing algorithms in locate data, but mantaining the structure is challenging with the frequent node joins. The most comon example is DHT that provides the capacity to map any given key to a node that is active at the moment.
In the second class we have semplicity in the implementation and the mantainance thanks to the lack of organization rules. Also is flexible to network changes. The routing is not efficient especially in large network.

Ex 2: Consider a distributed system constituted by n processes $\prod=\{p_1, p_2\dots p_N\}$ with unique identifiers that exchange messages through perfect point-to-point links and are structured in a ring topology (i.e., each process p_i can exchange messages only with processes $p_{(i+1) \bmod N}$ and stores its identifier in a local variable next).
Each process p_i knows the initial number of processes in the system (i.e., every process p_i knows the value of N).

1. Assuming that processes are not going to fail, write the pseudo-code of an algorithm that implements a (1, N) regular register.






```
upon event <rr, init> do
    val := ⊥
    next := P(i+1) mod N
    correct := Π

upon event <rr, read > do
    trigger < rr, readReturn | val >

upon event < rr, write | v > do
    val = v
    trigger < pp2pl send |[write, val]> to next
```

upon event < pp2pl Deliver | [write, val] > from next

if val >= v do

trigger < rr, write Return >