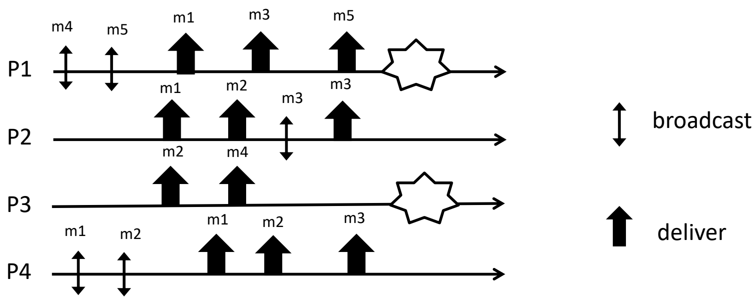
Dependable Distributed Systems
Master of Science in Engineering in Computer Science

AA 2022/2023

Lecture 23 – Exercises
November 23th, 2022
(Estimated time to complete all exercises: 3 hours)

Ex 1: Consider the execution depicted in the Figure

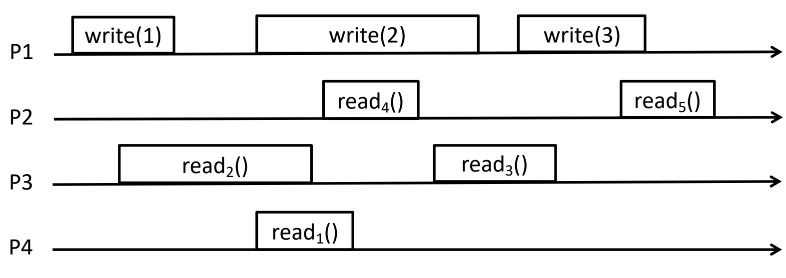


Answer to the following questions:

1. Which is the strongest TO specification satisfied by the proposed run? Motivate your answer.
2. Does the proposed execution satisfy Causal order Broadcast, FIFO Order Broadcast or none of them?
3. Modify the execution in order to satisfy TO(UA, WUTO) but not TO(UA, SUTO).
4. Modify the execution in order to satisfy TO(NUA, WNUTO) but not TO(UA, WNUTO).

NOTE: In order to solve point 3 and point 4 you can only add messages and/or failures.

Ex 2: 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 3: Consider the algorithm shown in the Figure

upon event ⟨ Init ⟩ **do**
    delivered := ∅; *pending* := ∅; *correct* := Π;
    forall m **do** *ack*[m] := ∅;
 	
upon event ⟨ *urb*, Broadcast | m ⟩ **do**
    pending := *pending* ∪ {(*self*, m)};
    trigger ⟨ *beb*, Broadcast | [DATA, *self*, m] ⟩;
 	
upon event ⟨ *beb*, Deliver | p, [DATA, s, m] ⟩ **do**
    ack[m] := *ack*[m] ∪ {p};
    if (s, m)/∈ *pending* **then**
      pending := *pending* ∪ {(s, m)};
      trigger ⟨ *beb*, Broadcast | [DATA, s, m] ⟩;
	upon event ⟨◊P,Suspect |p⟩**do**
	    correct := *correct* \ {p};
	
	upon event ⟨◊P,Restore |p⟩**do**
	    correct := *correct* ∪ {p};
	
	function candeliver(m) **returns** Boolean **is**
	    return (*correct* ⊆ *ack*[m]);
	
	upon exists (s, m) ∈ *pending* such that candeliver(m) **do**
	    delivered := *delivered* ∪ {m};
	    trigger ⟨ *urb*, Deliver | s, m ⟩;

Assuming that the algorithm is using a Best Effort Broadcast primitive and an Eventually Perfect Failure Detector ◊P discuss if the following properties are satisfied or not and motivate your answer

- *Validity*: If a correct process p broadcasts a message m, then p eventually delivers m.
- *No duplication*: No message is delivered more than once.
- *No creation*: If a process delivers a message m with sender s, then m was previously broadcast by process s.
- *Uniform agreement*: If a message m is delivered by some process (whether correct or faulty), then m is eventually delivered by every correct process.

Ex 4: 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 $p_{(i+1)\bmod N}$ 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.

Ex.5: Answer true or false to the following claims providing a motivation:

1. The performance of a system can be analyzed independently from its load.
2. Let us assume a single service with a single class of requests (i.e. a single workload component). If we are under the stability condition ($\lambda < \mu$) then the expected response time of the system is independent from the arrival pattern.
3. The workload parameters that mostly influence the performance of system are the arrival pattern and the service demands.
4. The time needed to run a simulation is upper-bounded by the simulated time
5. The reliability function R(t) associated to a component may increase after the restoration of a component

Ex.6: A service is provided through 3 types of components (e.g. database, webserver, and an application), referred with A, B and C. At least one instance of each component must be available to provide the service. One only instance is available for components A and C, whereas two instances of component B are available. Assuming that all failures and restorations are independent among them, that the failures rates of the three components A, B and C are respectively 0.2 , 0.5 and 0.3 faults per day, that the mean time to repair the components A, B, and C are respectively 2h, 3h, and 1.5h, is the service available at least 98% of the time? If not, is the availability target met deploying an additional instance of one of the components?

Ex.7: Let us assume a probabilistic version of the Eager Reliable Broadcast protocol in which every process retransmits a rb-delivered message only once with probability 0.5. The rb-broadcast operation is periodically triggered, and the calls follows an exponential distribution with average time between consecutive calls of 10 seconds. Assuming a distributed system of 30 processes and no processing delay, what is the minimum in-channel capacity per process (in terms of message per seconds) to ensure that the expected time for a rb-delivery is below 0.5 seconds?