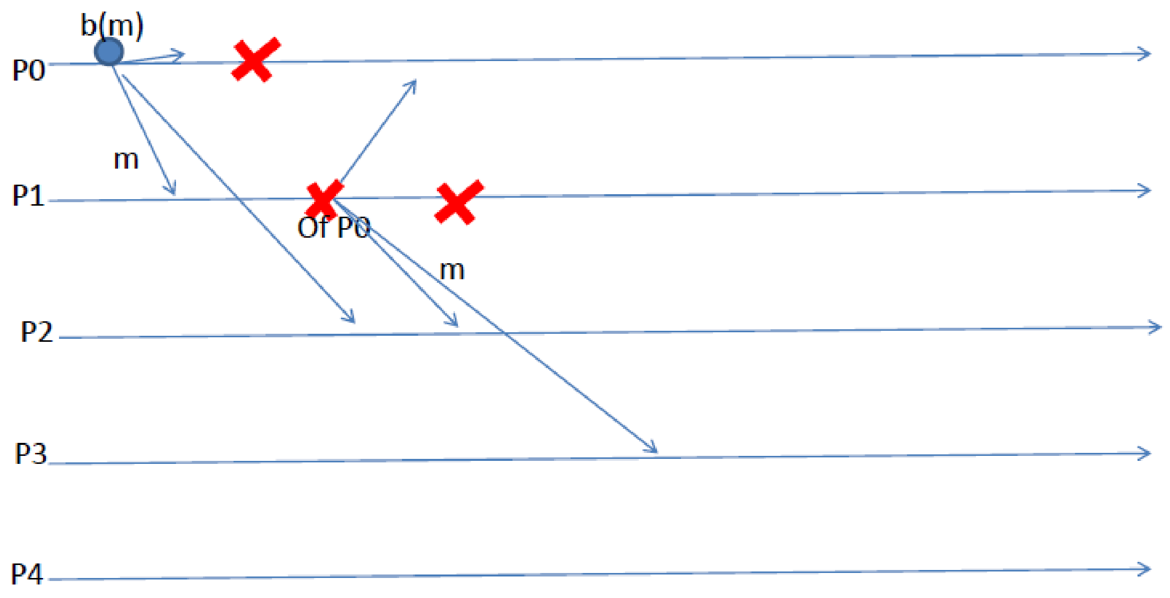
Distributed Algorithmics – TD4 Group Comm - M2 IFI, Ubinet-CSSR

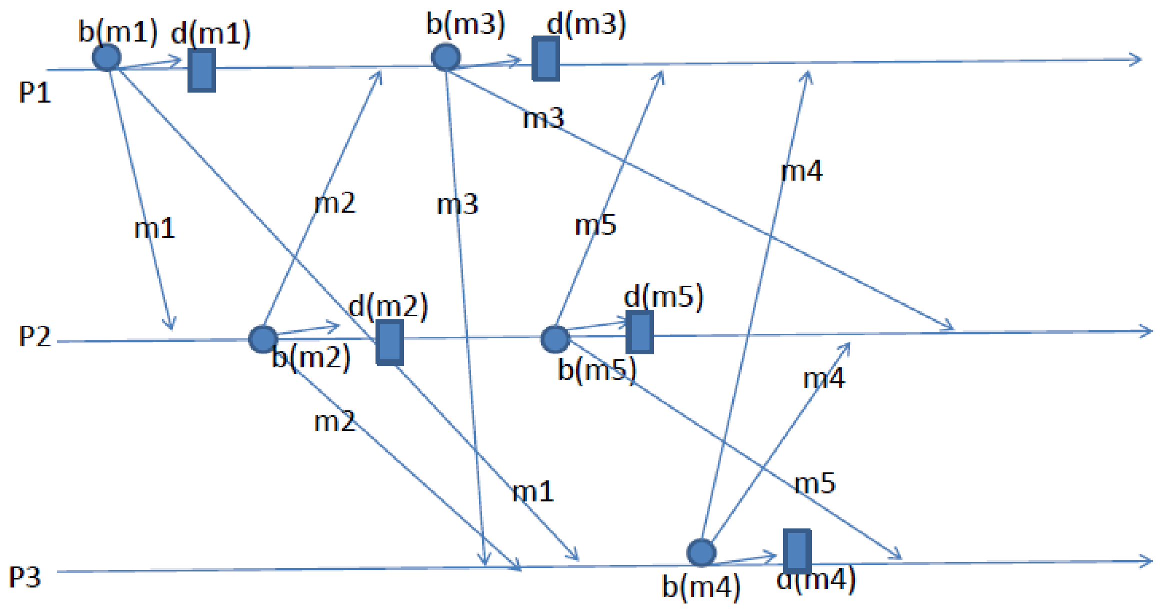
Exercice 1

Set up a nice example to illustrate the RBcast protocol behaviour, compared to Bebcast. You can assume that the presence of a Pefect Failure Detector allows you to detect accurately which are the failed processes.
Start from the following situation involving P0, P1…, P4. . P0 starts to broadcast message m, but fails after having sent the message to P1 and P2. P1 gets informed about the crash of P0 after it already received m; but, while it executes the corresponding code to relay the message m to all the others (assume it could relay it to P0, P2, P3 but not to P4). So for now, P4 has not yet a chance to receive m. However, due to the RBcast protocol, P4 will eventually receive it. You are asked to continue the scenario as you will, including at which moment each process gets informed by processes crashes, and at which moment the message m gets received



Exercice 2

We provide you with the following space-time diagram.



You are requested to give the precise ordering of messages m1, m2, m3, m4, m5 delivery on each of the 3 processes P1, P2, P3.
Explain why the traditional “merge” operation of vector clocks (which keep the maximum of the 2 compared vectors entries) does not need here to be made, and why only a “+1” operation on the entry V[j] on each process i is needed when Pi co-deliver the message originating from Pj.
Are the delivery orderings the same on all the processes ?

Exercice 3

Given the overall specification of the total order (reliable) broadcast protocol of the course, propose an algorithm based on the use of a sequencer. The sequencer itself could be one of the group members (i.e. the elected leader of the group). Obviously, this is coming to be an easy to implement solution, even if it features a central-bottleneck drawback.
Each message broadcast will be added an identifier, and a delivery order number. This last number is obtained through a request to the sequencer, which delivers numbers in consecutive increasing manner. Thanks to the obtained number, each process is capable to deliver received messages in a total ordered way.

Give the precise algorithm for both any process of the group, and the sequencer. Rely onto broadcasts only to interact from any process to/from the sequencer. Make sure to describe the algorithm using the module oriented notation.

Explain how the TO property gets fulfilled.
In case faulty processes (except the sequencer) could exist in the system, explain how it is the case that your complete algorithm ensures reliability (by which mean RB1-RB4 properties are ensured?)

Exhibit a possible execution of your algorithm in which total order of delivery is ensured, but not causal ordering for all broadcaster messages