DSM: Tutorial 5

Question 1

Improvement options:

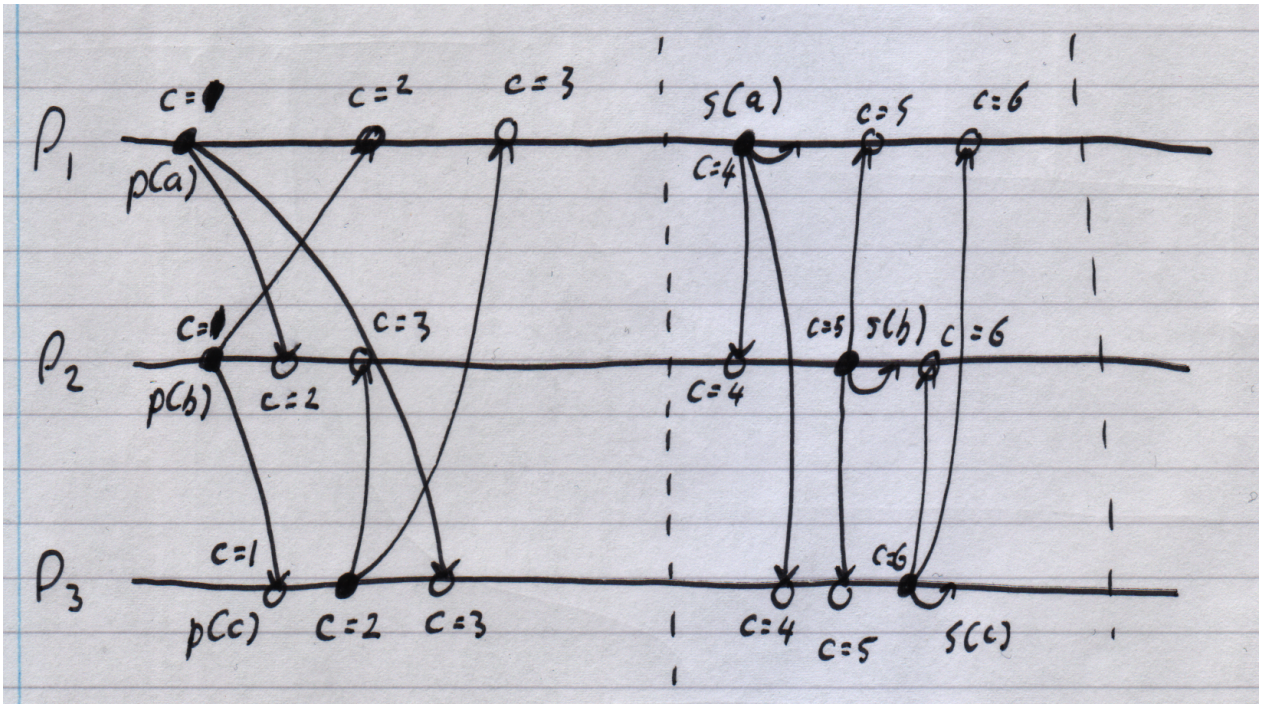
- Change the consistency ordering to reduce the blocking time, at the cost of lost requests.
- Don't bother with acknowledgements to reuce the blocking time, at the cost of lost requests.
- Allow a front-end to talk to back-ups to allow for load balancing, at the cost of consistency guarantees being lost.

Options for recovering from inconsistency:

- Detect inconsistency with checksums.
- Periodically copy the primary (may overwrite data).
- Merge differing states to capture all requests (may lose data).

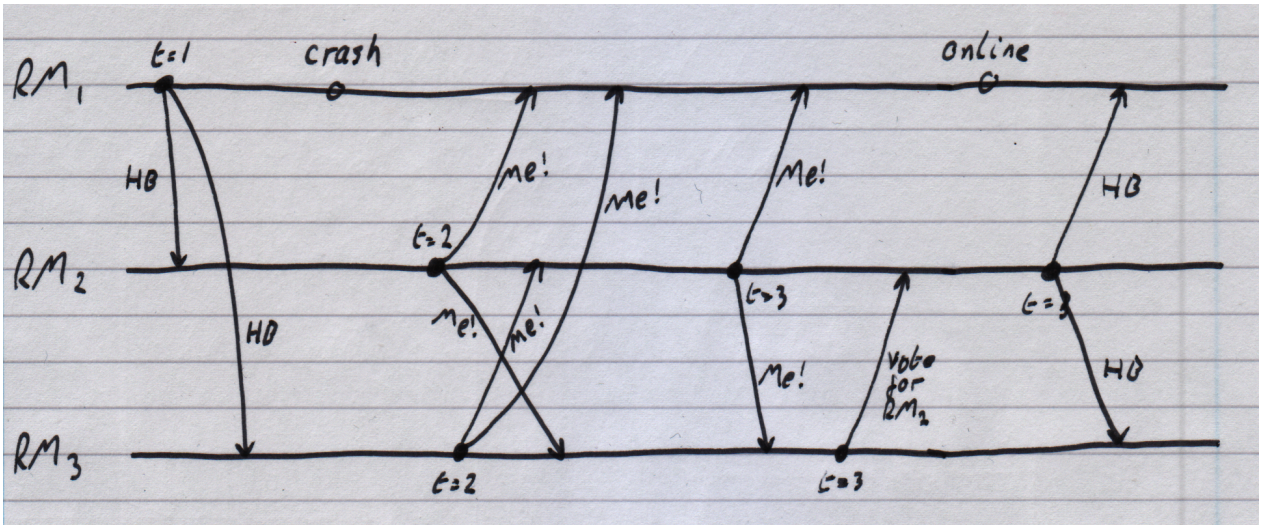
Question 2

In the first stage, an arbitrary messaging protocol is used for each process to propose one of `a`, `b` or `c`. In the second stage, each process seconds a value chosed from the ones it recevied, this time with total ordering. Each process decides on the first value it recevied, so `a` is decided.



To avoid bias, we need to make sure that no process has an advantage. Total ordering uses processor IDs to break ties when multiple messages have the same clock, which creates a bias. This could be mitigated by changing processor IDs periodically (which would also require consensus, but it doesn't need to be unbaised if the proposals are random).

Question 3



Any heartbeats from RM_1 when it comes back online will be ignored. When it receives a heartbeat with a higher term, it will know it is not the leader.

Question 4

A bug could introduce inconsistency at any point: incorrect/inconsistent parsing of input, failure to write data, etc. The inconsistency could be detected by comparing replicas (maybe using checksums) and overwriting one replica with data from another to achieve a consistent state (a process that may favour the newest copies of data).

Question 5

Primary Backup	**State Machine**
Requests are sent to one RM.	Requests are sent to all RMs.
When a backup RM fails, no pending requests should be lost. When the primary RM fails, an unbounded number will be lost during the failover period.	When one RM fails, no pending requests should be lost because other RMs will still receive the reqest.
For failstop errors, $t + 1$ RMs are needed for continually functioning service.	For failstop errors, $t + 1$ RMs are needed for continually functioning service.
Byzantine errors cannot be handled.	For Byzantine errors, $2t + 1$ RMs are needed for continually functioning service.