Spring 2025: Distributed Systems Midterm

March 13, 2025

Name: NetID:

Do not write in the boxes below.

1 (xx/15)	2 (xx/10)	3 (xx/15)	4 (xx/10)	Total (xx/50)

1 Short Questions [15 points]

a. State whether each of the properties listed below (taken from the Paxos paper or lecture) is a **safety** property or a **liveness** property. [$2 \times 3 = 6$ points]

Prop. 1. If a proposer proposes a value v, eventually the acceptors choose value (possibly $v' \neq v$).

Prop. 2. If a proposal with value v is chosen, then every higher-numbered proposal that is chosen has value v.

Prop. 3. An acceptor must accept the first proposal that it receives.

b. Your friend Rutherford tells you that he has created a RSM protocol that can tolerate up to f failures with more than $f+1$ nodes initially. It is a leader-based protocol, and for safety, when replicating an entry the leader requires responses from **all** nodes that have not yet failed. Is this protocol correct (i.e., does it maintain safety and ensure that entries are eventually committed if fewer than f nodes have failed) under the asynchronous model? Briefly (in 1–2 sentences) explain your answer. [4 points]

c. While working through an exam where you need to check that a history H is linearizabile (as you will in §2) you and your friend Rutherford arrive at two different total orders $<_y$ and $<_r$. Is it possible that both are valid linearizations of H, i.e., is it possible that $<_y \supseteq <_H$, $<_r \supseteq <_H$, and both $<_y$ and $<_r$ are consistent with the sequential specification? [5 points]

2 Linearize! [5×2 = 10 points]

Below we present execution histories from different data structures using the same notation as the Herlihy and Wing paper. As a reminder, histories are presented in the order in which events occur, invocations are written as:
`object operation process` (e.g., `p Enq(x) A` is process `A` invoking `Enq(x)` on object `p`)
and returns/responses are written as:
`object OK(value) process` (e.g., `p OK() A` is object `p` informing process `A` that the last call completed).

For each execution history, write down whether it is linearizable or not.

History 1. Object `p` is a first-in-first-out queue.

- `p Enq(1) A`
- `p Enq(2) B`
- `p Deq() C`
- `p OK(2) C`
- `p OK() A`
- `p OK() B`

History 2. Object `p` is a first-in-first-out queue.

- `p Enq(1) A`
- `p Enq(2) B`
- `p Deq() C`
- `p OK(2) C`
- `p OK() A`
- `p OK() B`
- `p Deq() B`
- `p OK(2) B`

History 3. Object `p` is a first-in-first-out queue.

- `p Enq(1) A`
- `p Deq() C`
- `p OK() A`
- `p Enq(2) B`
- `p OK(2) C`
- `p OK() B`

History 4. Object `p` is a first-in-first-out queue.

- `p Enq(1) A`

- p Enq(2) B
- p OK() A
- p Enq(3) A
- p OK() B
- p Deq() C
- p OK(2) C
- p Deq() C
- p OK(1) C
- p Deq() C
- p OK(3) C

History 5. Object p is a first-in-first-out queue.

- p Deq() C
- p Enq(1) A
- p OK(1) C
- p OK() A

3 Throw me a Raft [15 points]

3.1 Rumdar's Raft [7 points]

In the Raft paper (§5.4.1) and in class (Lecture 6) we talked about the requirement that a node *N* only votes for a candidate *C* if *C*'s log is at least as up-to-date as *N*'s (and of course if the term check passes). We said *C*'s log is at least as up-to-date if (a) the term for *C*'s last log entry is higher than *N*'s; or (b) both *C* and *N*'s last log entry is from the same term and *C*'s log is the same length or longer than *N*'s.

While implementing Raft for class, Rumdar misread this condition. On receiving a VoteRequest message for a term *t* from candidate *C*, a node *N* running Rumdar's Raft implementation (Raft-R from now on) votes for *C* if *t* is equal to or higher than the current term, *N* has not already voted for another node this term, and *C*'s log is *longer* than the node *N*'s log. Show an example run (or trace) where Raft-R violates State Machine Safety, or write an explanation for why such an erroneous run does not exist.

Note, you do not need to start your trace from the initial state, you should just show enough of trace to highlight the safety violation.

3.2 Wooden Tablets [$2 \times 4 = 8$ points]

While exploring New York you find a set of tablets that have been discarded by someone in anger. The tablets seem to have been created by writing down the state of each node in a 3-node Raft cluster. Unfortunately, only a part of the state was written down. Specifically, each tablet records:

(i) Current term.

(ii) Leader for the term.

(iii) Logs at each node (for brevity, in this question, we omit the actual command that is logged and only provide the log entry's term).

You want to reconstruct the state machine from these tablets, but to do so you need to identify which of the entries have been committed. For each of the tablets and log entries below mark whether they are:

1. Definitely committed [**DC**]: The entry was definitely committed, and can be safely applied.
2. Maybe committed [**MC**]: The tablet does not provide enough information to determine whether the entry was committed or not. As a result, it is unclear whether it has been applied or not.
3. Not committed [**NC**]: The entry has definitely not been committed.

Tablet 1. Current term: 3. Current Leader: A

Node	**1**	**2**	**3**	**4**	**5**
A	1	1	2	2	3
B	1	1	2	2	
C	1	1	2	2	
Answer					

Tablet 2. Current term: 3. Current Leader: A

Node	**1**	**2**	**3**	**4**	**5**
A	1	1	2	2	2
B	1	1			
C	1	1	2	2	2
Answer					

Tablet 3. Current term: 5. Current Leader: A

Node	**1**	**2**	**3**	**4**	**5**
A	1	2	3	3	3
B	1	2	4	4	
C	1				
Answer					

Tablet 4. Current term: 5. Current Leader: A

Node	**1**	**2**	**3**	**4**	**5**
A	1	2	3	5	5
B	1	2	4	4	
C	1	2	3	5	
Answer					

4 Do Things Change in (Multi)Paxos? [10 points]

4.1 Wooden Tablets on an Island [$1 \times 4 = 4$ points]

This question is trying to get you to compare Raft and Multipaxos. We will do the same exercise as the question in §3.2: you will look at a few tablets for which you know the current leader and view ID/ballot (see lecture 7). Unlike §3.2, the logs in this case specify the value (or command) at each index. Your goal is similar: for each log entry mark whether it is

1. Definitely committed [**DC**]: The entry was definitely committed, and can be safely applied.
2. Maybe committed [**MC**]: The tablet does not provide enough information to determine whether the entry was committed or not. As a result, it is unclear whether it has been applied or not.
3. Not committed [**NC**]: The entry has definitely not been committed.

Tablet 1. Current view ID/ballot: 3. Current Leader: A

Node	**1**	**2**	**3**	**4**	**5**
A	*a*	*b*	*c*	*d*	*e*
B	*a*	*b*	*c*	*d*	
C	*a*	*b*	*c*	*d*	
Answer					

Tablet 2. Current view ID/ballot: 3. Current Leader: A

Node	**1**	**2**	**3**	**4**	**5**
A	a	b	c	d	e
B	a	b			
C	a	b	c	d	e
Answer					

Tablet 3. Current view ID/ballot: 5. Current Leader: A

Node	**1**	**2**	**3**	**4**	**5**
A	a	b	c	e	g
B	a	b	d	f	
C	a				
Answer					

Tablet 4. Current view ID/ballot: 5. Current Leader: A

Node	**1**	**2**	**3**	**4**	**5**
A	a	b	c	f	f
B	a	b	d	e	
C	a	b	c	f	
Answer					

4.2 Duplicate Proposal Numbers? [6 points]

In class, when discussing the Synod we were careful about using unique proposal numbers, that is ensuring that no two proposers used the same proposal ID. Can a safety property be violated if two proposers (say proposer A and B) use the same proposal ID (say p)? If a safety property can be violated, show an example run (or trace) where this happens (you only need to show enough to illustrate the problem). If no safety property is violated, briefly explain why this is the case.

5 Survey

This is just to get information about your experience in class and is not graded.

a. How is the pacing of the class? Do you think we should be going slower, faster, etc.

b. Do you have any concerns about the lectures, specifically in terms of what is covered, the amount of material covered each lecture, or their relationship to the readings.

c. Do you have any concerns about the labs: specifically do you feel they are either too difficult or too simple?

d. Do you feel that Elixir has been an impediment to your work on labs? If so, what specifically has proven to be a problem?

THE END — Rest of the Pages are Scratch Space