Homework 3

CS425/ECE428 Spring 2024

Due: Wednesday, March 27 at 11:59 p.m.

1. **Paxos 1**

   (a) (2 points) A software engineer at Goggle is asked to implement Paxos algorithm for achieving consensus among N servers. He implements Phase 1 of Paxos correctly, where the proposer sends a *prepare* message to majority of acceptors, and the acceptors respond as per the Paxos specification. However, he leaves a bug in his implementation for Phase 2, allowing the proposer to send an *accept request* after receiving an OK from *any one* of the acceptors. (He still sends the accept request to *all* the acceptors after receiving this one OK.) Explain, using an example, how this may result in a safety violation.

   (b) (1 point) When should the proposer send its *accept* request to avoid the safety violation above?

   (c) (2 points) After the necessary bug fixes in how the two phases of the algorithm are implemented, the engineer decides to do some performance optimizations. He noticed that the acceptor's code (that he copied from someone else) reads and writes the accepted value and the corresponding proposal number from a file kept in persistent storage, which is clearly slower than reading/writing from a program variable stored in volatile RAM. He therefore makes the performance optimization, allowing the acceptors to read/write from a variable in RAM instead of a file in the disk. Can this clever-seeming optimization result in a safety violation? If yes, how?

   (d) (1 point) After fixing all his bugs, and thoroughly testing his code, he is confident that his Paxos implementation will result in no safety violation. However, he realized that his implementation still does not guarantee liveness. He is now working on fixing that. Do you think he will be successful in this endeavor?

2. **Paxos 2**

   Consider a system of five processes that implement the Paxos algorithm for consensus. As shown in Figure 1, P1 and P2 are proposers. P3, P4, P5 are acceptors. P1 sends a *prepare* message with proposal number 3 to processes P3 and P4, receives their replies, and sends an *accept* message with proposed value of 12 for proposal #3. P2 concurrently sends a *prepare* message with proposal #6 to P4 and P5, with an initial intention to propose a value of 15 if it receives sufficient replies. Only a subset of responses from processes P3, P4, and P5 are shown in the figure. Assume no other proposals are initiated. Answer the following sub-questions.

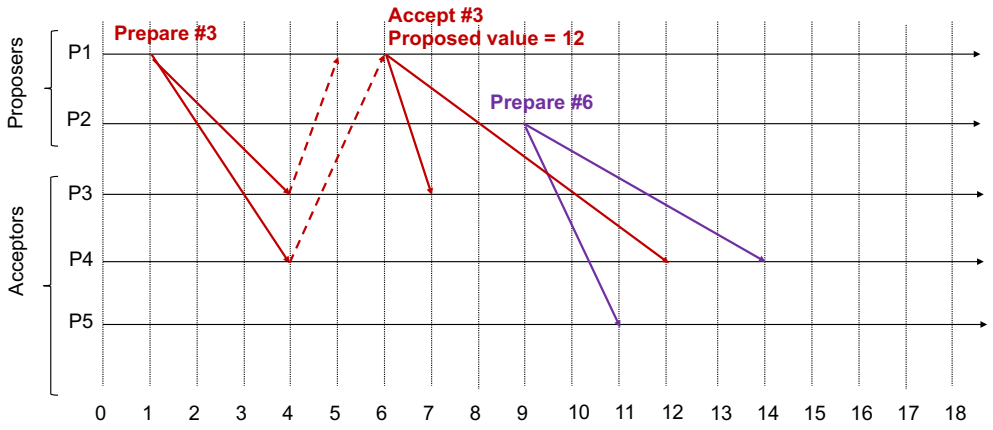

Figure 1: Figure for question 2

   (a) (1 point) Which processes will *accept* P1's proposal?

(b) (1 point) Which processes will reply back to P2's *prepare* message? Also specify, what value (if any) will the reply contain.

(c) (1 point) Will P2 send out an *accept* message for its proposal #6? If yes, what will be the corresponding proposed value?

(d) (3 points) Consider the state of the system at time 8 units. Has the value 12 proposed by proposal #3 been implicitly decided upon at this point? If yes, explain why? If not, construct a possible scenario that may occur after time 8 units where the system ends up deciding on a different proposed value. The scenario should involve a temporal sequence of events that can modified from the one shown in the figure beyond time t=8units but must comply with what is shown until t=8units.

(e) (3 points) Now instead, consider the state of the system at time 13 units. Has the value 12 proposed by proposal #3 been implicitly decided upon at this point? If yes, explain why? If not, construct a possible scenario that may occur after time 13 units where the system ends up deciding on a different proposed value. The scenario should involve a temporal sequence of events that can modified from the one shown in the figure beyond time t=13units but must comply with what is shown until t=13units.

3. **RAFT Leader Election**

Consider a system of 5 processes $\{P1, P2, P3, P4, P5\}$ using Raft's algorithm for leader election. Suppose P1, the leader for term 1, fails and its four followers receive its last heartbeat at exactly the same time. Answer the following questions assuming that the election timeout is chosen uniformly at random from the range [100,500] ms (unless otherwise specified), no processing delay exists, and the one-way delay for all messages between two processes are as shown in Figure 2. The processes communicate with one-another only through their direct channels (not via other processes). Each question below is independent of others.

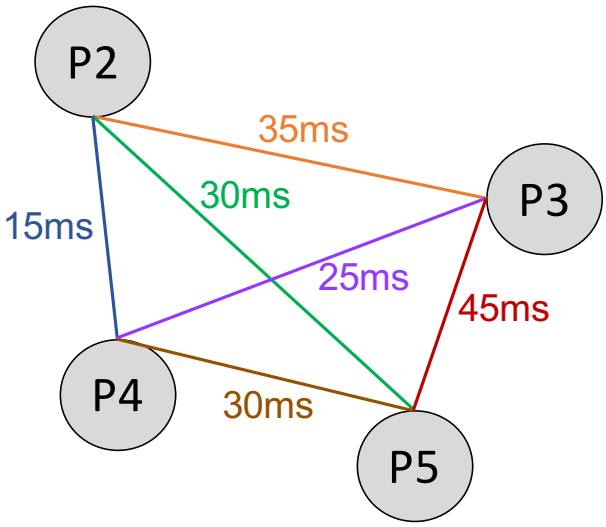

Figure 2: Figure for question 3

(a) (2 points) Suppose P2 and P3 both set their election timeout to 150 ms and call an election for term 2. Assume P4 and P5 have their timeout values set to more than 400 ms. Which candidate (P2 or P3) will each of the four alive processes vote for? Will a leader be elected for term 2? If yes, which process?

(b) (2 points) Suppose P3 sets its election timeout to 150 ms and calls an election for term 2, and P2 sets its timeout to 162 ms and also calls an election for term 2. Assume P4 and P5 have their timeout values set to more than 400 ms. Which candidate (P2 or P3) will each of the four alive processes vote for? Will a leader be elected for term 2? If yes, which process?

(c) (6 points) Suppose P3 sets its election timeout to 150 ms and calls for an election for term 2. Assume P4 and P5 have their timeout values set to more than 400 ms. What range of timeout values for P2 (within [100,500] ms) will certainly result in:

(i) P2 winning the election for term 2?

(ii) P3 winning the election for term 2?

(iii) split vote for term 2?

4. **RAFT Log Consensus**

Consider a system of three servers $\{S_1, S_2, S_3\}$ wanting to achieve log consensus using the Raft algorithm. For each sub-part below, state whether the shown snapshot of log entries at each server could arise from a valid run of the Raft algorithm. If yes, construct a scenario that would lead to these log entries in Raft's execution. If not, explain what makes the entries invalid.

Each number in the shown log entries represents the Raft term that the corresponding event is associated with.

For the valid log entries, the scenario you construct should include, for each term: which server gets elected as the leader, which servers vote for it, and which log entries does it append / replicate at each server.

(a) (3 points)

S_1: 1, 2
S_2: 1, 1
S_3: 1, 2

(b) (3 points)

S_1: 1, 1
S_2: 1, 1, 2
S_3: 1, 1, 3

(c) (3 points)

S_1: 1, 1, 4
S_2: 1, 1, 2
S_3: 1, 3

(d) (3 points)

S_1: 1, 1, 3, 3
S_2: 1, 2, 2
S_3: 1, 1, 3

(e) (3 points)

S_1: 1, 1, 2
S_2: 1, 2, 2, 2
S_3: 1, 2, 2