Questions from Past Exams

Distributed Computing

Question 1

Consider a synchronous path network connecting n processors, $v_1, \ldots, v_n$. The network is unreliable, and messages might get lost. The sender of a message does not know whether it was lost or not.

Let $L(i,t)$ denote the event that the message sent from v_i to v_{i+1} on round t was lost. Assume that the probability of this event is some fixed $0 < p < 1$, and that the events $L(i,t)$ are all independent.

The following algorithm is proposed for broadcasting a message from v_1:
- The source v_1 sends the message repeatedly to v_2 for B times, on rounds $1, \ldots, B$, and then halts.
- Whenever a processor v_i receives the message for the first time on some round t, it sends the message repeatedly to v_{i+1} for B times, on rounds $t+1, \ldots, t+B$, and then halts.

(a) Give an upper bound on the *maximum* running time of the algorithm in the worst case.

(b) Express the probability that the message will successfully reach all the processors as a function of p, B and n.

(c) Determine the smallest value of B guaranteeing that this probability is at least $1 - 1/n$.

(d) Give an upper bound on the *expected* running time of the algorithm.

Question 2

Consider 7 processors $p_1, \ldots, p_7$ connected by a complete network. Consider a generic synchronous protocol π that operates for two rounds, where in each round, each processor sends everything it knows to all other processors. Let α and β be two executions of π that satisfy the following conditions:

(a) Each processor p_i $(1 \leq i \leq 7)$ starts executions α and β with the same inputs.

(b) In each of the two executions, at most one processor has failed (by crashing).

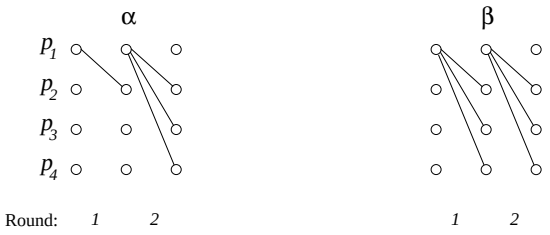
(c) The two executions are *different*.

(d) Processor p_1 did not fail in either execution, and $\alpha \overset{p_1}{\sim} \beta$.

Prove that under these assumptions, no processor has failed in the first round of either α or β.

Question 3

Let Π be a synchronous protocol operating in two rounds on a complete network, where in each round, every processor tells all other processors everything it knows. Consider the two executions α and β of Π depicted in the figure below, where an edge connecting p_i to p_j means that p_i failed to send a message to p_j in this round.



Prove or disprove each of the following two claims.

(a) $\alpha \underset{[1]}{\approx} \beta$.

(b) $\alpha \underset{[2]}{\approx} \beta$.

Question 4

Consider the following two strengthened variants of the Vote primitive presented in class as part of the randomized agreement protocol. For each of these variants, describe a deterministic algorithm solving it in an asynchronous model with up to t stop failures and prove its correctness, or prove that such an algorithm does not exist.

Vote_A Replace property (V1) with the requirement that all nonfaulty processors decide on the *same* value out of the five possibilities.

Vote_B There are only *three* possible outcomes, namely, $0, \perp, 1$.
Property (V1) still requires that all nonfaulty processors decide on one of two consecutive values out of the three.
Property (V2) requires that if all nonfaulty processors have input a then all nonfaulty processors decide on a.

Question 5

In each of the following cases, determine which of the three components of the asynchronous randomized agreement protocol for crash failures studied in class (namely, Broadcast, Voting and Global coin flip) might fail, and explain how (by providing a failing scenario).

1. Byzantine failures are allowed
2. $t < n \leq 3t$
3. The adversary is non-oblivious, i.e., it may make its future scheduling decisions based on the outcome of previous random choices made by the participants.

Question 6

Consider the following generalization of the coordinated attack problem. There are n generals $p_1, \ldots, p_n$ participating in the attack. The communication between them is based on a given graph G (each general can exchange messages only with its neighbors on G). Communication is synchronous but unreliable (messages might get lost). Each general p_i has an input bit $x_i \in \{0, 1\}$, and it must decide on an output bit $y_i \in \{0, 1\}$. The problem requirements are as follows.

Agreement: All output bits are the same.

Validity: If $x_i = 0$ for every i and all sent messages were delivered, then $y_i = 0$ for every i.
If $x_i = 1$ for every i and all sent messages were delivered, then $y_i = 1$ for every i.

Termination: All generals reach decision within finite time.

Prove that there is no deterministic algorithm for solving the problem.

Question 7

Consider the following token-based algorithm for mutual exclusion on an n-node ring, $n \geq 3$. The nodes $v_0, \ldots, v_{n-1}$ have binary states $S_0, \ldots, S_{n-1} \in \{0, 1\}$. Each node sees its own state and the states of its two neighbors on the ring. The rules for holding and releasing the token are as follows.

1. The node v_0 holds the token if $S_0 = S_{n-1}$.
2. The node v_{n-1} holds the token if $S_0 = S_{n-2} \neq S_{n-1}$.
3. For $1 \leq i \leq n-2$, the node v_i holds the token if $S_{i-1} \neq S_i = S_{i+1}$.

The node v_i releases the token by flipping its state bit S_i.

a. Prove that in every configuration, at least one node holds a token.

b. Prove that the algorithm is a valid mutual exclusion algorithm if started from an appropriate configuration in a failure free setting. Namely, show that in such setting, there is an initial configuration C_0 such that starting from C_0, the algorithm cycles through a periodic sequence of configurations, granting tokens to the nodes in a fair (round-robin) manner, with exactly one node holding a token in each configuration.

c. Prove that the algorithm is not self-stabilizing.