Homework 4

CS425/ECE428 Spring 2023

Due: Wednesday, April 10 at 11:59 p.m.

1. **Blockchains**

   In a system using a blockchain for distributed consensus, in order to add a block to a chain, a participating node must solve the following puzzle: it must find a value x such that its hash, $H(x||seed)$, is less than T. The hash function is such that a given value of x can uniformly map to any integer in $[0, 2^{256} - 1]$. Assume T is set to 2^{216}.

   (a) (2 points) What is the probability that a given value of x, randomly chosen by the participating node, is a winning solution to the puzzle (i.e. $H(x||seed) < T$)?

   (b) (2 points) Assume a participating node adopts the standard strategy for solving the puzzle: it randomly picks a value x and checks if it is the winning solution. It keeps repeating this step, until a winning solution is found. Further assume that, for simplicity, the strategy is memoryless (unoptimized), in the sense that a value of x that has already been checked can get re-checked if it is randomly selected again. If the node can compute and check hashes at the rate of 2^{17} values per second, what is the probability of finding a winning solution within 3 hours? (You may round your answer to five decimal places.)

   (c) (2 points) Assume that of the 5000 participating nodes in the system. Of these, 1000 nodes can compute and check hashes at the rate of 2^{17} values per second (lets call these set of nodes Group A). The remaining 4000 nodes can compute and check hashes at the rate of 2^{10} values per second (lets call these set of nodes Group B). Each node starts solving the puzzle at exactly the same time. What is the probability that at least one node in the system finds a winning solution in 3 hours? (You may round your answer to four decimal places.)

   (d) (2 points) What is the probability that a winning solution is found within 3 hours by a node in Group A and not in Group B, where Group A and Group B node groups are as described in part c? (You may round your answer to four decimal places.)

   (e) (2 points) What is the probability that a winning solution is found within 3 hours by a node in Group B and not in Group A, where Group A and Group B node groups are as described in part c? (You may round your answer to four decimal places.)

2. **Concurrency: Two-phase locking, Deadlocks, and Timestamped Ordering**

   Consider the following two transactions, each with five operations:

	$T1$	$T2$
1	write A	read C
2	read B	read B
3	write B	read A
4	write D	write D
5	read A	read D

   (a) (2 points) Write down all the conflicting pairs of operations across the two transactions. (You can refer to each operation as $Tn.m$; e.g., $T1.1$ is "write A", $T1.2$ is "read B", and so on).

   (b) (3 points) Is the following interleaving of operations across $T1$ and $T2$ serially equivalent? Explain why or why not.

$T1$	$T2$
write A	
	read C
	read B
read B	
write B	
	read A
	write D
write D	
read A	
	read D

(c) (4 points) Is there a *non-serial* interleaving of operations across $T1$ and $T2$, that could result from using strict two-phase locking (with read-write locks), and is equivalent in effect to a serial execution of $T1$ followed by $T2$? If yes, provide an example. If not, explain why.

(d) (4 points) Is there a *non-serial* interleaving of operations across $T1$ and $T2$, that could result from using strict two-phase locking (with read-write locks), and is equivalent in effect to a serial execution of $T2$ followed by $T1$? If yes, provide an example. If not, explain why.

(e) (4 points) Write down a partial interleaving of the operations across $T1$ and $T2$ that is compliant with strict two-phase locking (with read-write locks) and leads to a deadlock. List which lock (and in which mode – read or write) will be waited upon by each transaction in your deadlock.

(f) (4 points) Write down an interleaving of the operations across $T1$ and $T2$ that is serially equivalent, but impossible with strict two-phase locking. Explain what makes the interleaving impossible with strict two-phase locking.

(g) (4 points) Write down a partial interleaving of the operations across $T1$ and $T2$, that could result from using *timestamped ordering* and would cause one of the transactions to be aborted. Clearly indicate which timestamp value (picked between 1 or 2) gets assigned to each transaction in your example. Also mention the relevant state maintained by the objects for timestamped ordering. You can use the example format like "write X (X.committedTS = 1, X.RTS = [1,2], X.TW=[2])" to indicate the state maintained by the object (i.e. timestamps for reads and tentative writes) after an operation has been executed.

3. **Two-Phase Commit**

Figure 1 shows a system of three servers processing a distributed transaction. Server 1 is the coordinator and interacts with the client. The network delay between the client and the coordinator, and among the three servers is indicated in the figure.

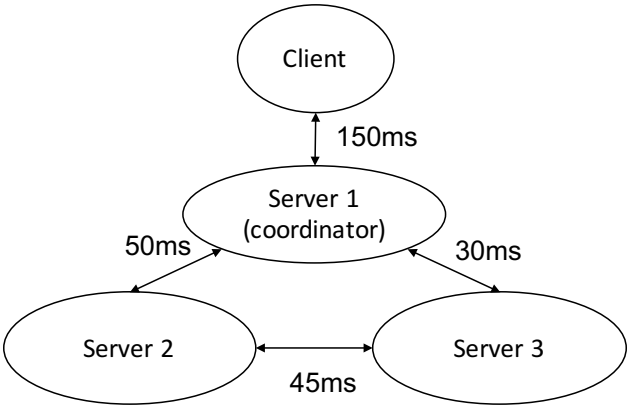

Figure 1: Figure for question 3

Any local processing at a server or self-messages take negligible time. The client issues a COMMIT request for its transaction at time t=0s. Assuming no messages are lost, no server crashes, and no server wishes to abort the transaction. Answer the following questions:

(a) (3 points) When will each of the three servers locally commit the transaction?

(b) (2 points) What is the earliest time at which the coordinator can safely send a message to the client stating that the transaction will be successfully committed?