Homework 5

CS425/ECE428 Spring 2021

Due: Thursday, April 15 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^{230}.

   (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) (3 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 hash and check 2^{10} values per second, what is the probability of finding a winning solution within 5 hours? (You may round your answer to two decimal places.)

   (c) (3 points) Assume there are 500 participating nodes in the system, and that each node starts solving the puzzle at exactly the same time. Assuming the same rate of computing hashes at each node (i.e. 2^{10} values per second), what is the probability that at least one node in the system finds a winning solution in 10 minutes? (You may round your answer to two decimal places.)

2. **Transaction Processing**

   A bank uses a transaction processing system that complies with ACID properties. Within each transaction, a user can issue one or more of the following operations: (i) `DEPOSIT <account> <amount>` which deposits the specified amount into the specified account, (ii) `WITHDRAW <account> <amount>` which withraws the specified amount from the specified account, and (iii) `BALANCE <account>` which immediately displays the current balance in the specified account (also including the effects of operations previously executed within the same transaction). As a consistency check, if at the *end of a transaction*, any account has a negative balance, the system aborts that transaction.

   Consider the five transactions shown below that are executed serially (one after another) in order T1, T2, T3, T4, T5. Answer the following questions, assuming all accounts referred in the transactions have a balance of zero before T1 is executed.

   ```
   T1:  DEPOSIT A 20; DEPOSIT B 30; DEPOSIT C 50
   T2:  DEPOSIT A 40; WITHDRAW B 50; DEPOSIT C 20; BALANCE A
   T3:  WITHDRAW A 10; DEPOSIT B 20; WITHDRAW C 60; BALANCE B
   T4:  WITHDRAW A 15; DEPOSIT B 25; WITHDRAW C 10
   T5:  BALANCE A; BALANCE B; BALANCE C
   ```

   (a) (5 points) For each transaction, state whether it gets committed or aborted.

   (b) (5 points) What will be the result displayed by each of the `BALANCE` operations invoked in the transactions?

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

   Consider the following two transactions, each with five operations:

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

(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., $T2.1$ is "read C", $T2.2$ is "write D", and so on).

(b) (2 points) Is the following interleaving of operations across $T1$ and $T2$ serially equivalent? Explain why or why not.

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

(c) (4 points) Write down 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 $T2$.

(d) (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.

(e) (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.

(f) (3 points) Write down a partial interleaving of the operations across $T1$ and $T2$, that could result from using *timestamped ordering*, and leads to $T1$ getting aborted.

(g) (3 points) Write down a *non-serial* interleaving of operations across $T1$ and $T2$, that could result from using *timestamped ordering*, and is equivalent in effect to a serial execution of $T1$ followed by $T2$. It should not result in any transaction getting aborted.