CS425 Fall 2023 – Homework 4

(a.k.a. "The Interview")

Out: Nov 1, 2023. Due: Dec 1, 2023 (2 pm US Central time.)

Topics: Lecture 19 and onwards (RPCs, Concurrency Control, Replication Control, and the rest of the topics)

Instructions:

1. **Attempt any 8 out of the 10 problems** in this homework (regardless of how many credits you're taking the course for). Please do not attempt more than 8. If you attempt more, that creates more work for TAs, so we will take the **lowest** 8 scores among your questions. Choose 8 wisely!
2. Please hand in **solutions that are typed** (you may use your favorite word processor. We will not accept handwritten solutions. Figures (e.g., timeline questions) and equations (if any) may be drawn by hand (and scanned).
3. **All students (On-campus and Online/Coursera)** – Please submit PDF only! Please submit on Gradescope. [https://www.gradescope.com/]
4. Please **start each problem on a fresh page**, and **type your name at the top of each page**. **And on Gradescope please tag each page with the problem number!**
5. Homeworks will be **due at time and date noted above. No extensions. For DRES students only:** once the solutions are posted (typically a few hours after the HW is due), subsequent submissions will get a zero**. All non-DRES students must submit by the deadline time+date.**
6. Each problem has the same grade value as the others (10 points each).
7. Unless otherwise specified in the question, you cannot use the Web for the question. By default, the only resources you can avail of in your HWs are the provided course materials (slides, textbooks, etc.), and communication with instructor/TA via discussion forum and e-mail.
8. You can discuss lecture concepts and the questions on Piazza and with your friends, but you cannot discuss solutions or ideas on Piazza.

Prologue: After the success of your epic Saturn mission and your celebrated return to Earth, you receive many offers of employment at top companies and universities. But before you get hired, you have to go through interviews at these places. Your goal is to attempt 8 interviews and ace them so that you have the maximum choice of where you want to (After all, if you're attached, the choice will likely be made by your significant other, so you want to give her/him the most choice! If you're not attached, you want to have the most choice anyway, right? ;).

The storylines, statements, events, things, and games in this homework are purely fictitious. Any resemblance to persons, places, or events, living or dead, past, present, or future, is purely coincidental. All actions, words, and thoughts, ascribed to real entities like people and companies, are purely fictitious.

Problems:

1. (For this question you can use the Web. However, please write answers in your own words! Don't cut and paste from websites.) You put "Consensus" on your resume, and immediately every single cryptocurrency company on this planet wants to interview you. So you pick the one with the highest market value – you're going to be a gazillionaire! But before you go to the interview, you need to educate yourself, especially as your Professor Indy seemed to not think too highly of blockchains. Answer each of the following (Please limit your answer for each part to less than 50 words. Be concise!)
    a. What is the difference between a blockchain and a cryptocurrency like Bitcoin?
    b. What is "mining" and "proof of work"?
    c. What is the key difference between "permissioned" and "permissionless" blockchains?
    d. Does Bitcoin use a consensus algorithm like Paxos or Raft? If yes, say how. If no, say briefly what Bitcoin's consensus algorithm looks like.
2. During an interview at IBM Research Almaden, they tell you that the relational database model was invented by E. F. Codd, so they love transactions. They give you the following transactions
    - T1: write(a, caz, T1); read(b, T1); write(c, foo, T1);
    - T2: write(a, bar, T2); read(b, T2); write(c, baz, T2);

   They ask you
    i. If one interleaves the transaction's statements alternately, starting with T1's first statement, then T2's first statement, then T1's second statement, and so on – is this interleaving serially equivalent? Say why.
    ii. Your interviewer claims *any* interleaving of T1 and T2 is serially equivalent. If you agree, prove it. If not, give a counter-example.
    iii. Remove the last statement of each of T1 and T2 (accesses to c) to derive T1′ and T2′. How many of *all possible* interleavings of T1′ and T2′ are serially equivalent and how many are not? (there should be 6 total interleavings) Justify your answer.

3. Your next interviewer, Montgomery Burns comes up with an “excellent” scheme. He tells you that they use a transaction management system that uses 2 phase locking where all locks are acquired at the beginning of the transaction (if a lock attempt blocks, the transaction blocks and does not abort), and all locks are released at the commit point. He tells you that the system restricts transactions to acquire locks only in lexicographically *increasing* order of an object’s “special field”. Burns claims that this system will not deadlock. In each of the following cases, say whether the above system will deadlock or not (give a proof or a counter-example).
    a. Each object’s “special field” is set to access order of the transaction, i.e., a transaction T locks objects in the order that T will first access these objects (note that all locks are acquired at T’s start point).
    b. Each object’s “special field” is the object’s ID (which is globally unique).
    c. Each object’s “special field” is the unique ID (UTC time, with a random salt/nonce appended) of the object’s first creation time.
4. Your next interview is with Gordon Gekko, who thinks greed is good and who wants to one-up Montgomery Burns. Gekko presents you a transaction system with “greedy, just in time” locking (only one lock mode – no read or write locks). In this scheme he has each transaction acquiring a lock for an object just in time, right before each access to that object O, but then the transaction holds on to the lock until after the last access to the object O – after this last access to object O, the lock on O is immediately released. Assuming only runs occur where there are no deadlocks, will this system satisfy serial equivalence in such (deadlock-free) runs? Present a proof/counterexample, as appropriate.
5. (For this question only, you can only look up only the linked paper) The folks at Berkeley and Stanford are surprised to know that you know about their invention DRF. In lecture we discussed Dominant Resource Fairness (DRF), but we did not discuss equations to derive “fair” allocations. Look at the equations in the original paper, especially Section 4 and 4.1 (only). Here is the only paper you can access for this question: https://courses.engr.illinois.edu/cs425/fa2023/nsdi_drf.pdf . Then calculate fair allocations for each of the following cases (Cloud has 20 CPUs, 40 GB RAM). Please only provide answers that have only integer values for number of tasks and number of CPUs assigned to each (memory can be fractions of GB).
    a. Job 1’s tasks: 1 CPUs, 1 GB. Job 2’s tasks: 2 CPUs, 2 GB.
    b. Job 1’s tasks: 2 CPUs, 4 GB. Job 2’s tasks: 2 CPUs, 4 GB.
    c. Job 1’s tasks: 2 CPUs, 1 GB. Job 2’s tasks: 4 CPUs, 8 GB.
    d. Job 1’s tasks: 2 CPUs, 8 GB. Job 2’s tasks: 6 CPUs, 2 GB.

6. (Just to be clear -- You cannot use the Web for this question.) You also interview for a research position at Berkeley. At Berkeley, where they invented the Mica Mote (true fact!), they say they are building a sensor network in the ocean to monitor movements of whales (and ensure that none of them are killed for sport by a Trump or a Moby Dick). The sensors measure speed, but they are also mobile sensors because the whales are continuously swimming. The deployment spreads 3000 MICA motes over several thousands of square miles. They would like to get hourly measurements of speed metrics across all sensors (the data does not need to be seen immediately, but eventually they will need all hours). The base station is on shore. They want to use a spanning tree among the whale sensors, with the base station as the root. What *minimal* data would you collect via the intermediate nodes of the tree for the aggregation if you were trying to calculate each of the following (i.e., what is the partial aggregation function? And what is the exact data you would pass along?). You can assume each whale provides one measurement data item per hour (i.e., speed, ID). (each part below is a separate scenario). You can also assume each intermediate node receives all its children's data (you need to decide what this data is), calculates its result via your partial aggregation function, and then passes the result to its parent. Please limit to answer to each part to under 50 words. You don't need to write pseudocode, but please be clear and unambiguous.
    a. Highest and lowest IDs of currently active whales.
    b. Speeds and IDs of the top 5 fastest whales.
    c. Count of all whales (currently in the system).
    d. Average speed across all whales.
    e. 75^{th} percentile value of speed across all whales.
7. For this question you can look up the Web. While you're interviewing with Cruella De Ville, she is pensive. She is wondering about the widely used "MOESI" and "MESI" (not Messi, who's of course the GOAT!) protocols for cache coherence. In less than total 100 words, say 1) what the key difference is between these two standards, and 2) what is the relationship between MOESI and MESI on the one hand, and the invalidate and update protocols discussed in lecture on the other hand.
8. During your exciting interview at MIT, you find that they seem to like distributed shared memory. They ask you the following question involving 5 processes P1-P5 in a distributed shared memory system using invalidate. Process P3 wants to write a page. In each of the following cases, say what is the series of operations that needs to happen for P3 to be able to write (warning: there be tricks below!) -- You cannot use the Web for this question. If the setup seems wrong to you, you should point out ALL errors in it.

   a. P4 is the owner and is holding the page in a Write mode. P2 is also holding the same page but in the Read mode.
   b. P3 is owner and holding the page in a read mode. P1 and P2 are each holding the page in a Write mode.
   c. P1, P2, P3 are each holding the page in a Read mode, and P3 is the owner.
   d. P1, P2 and P3 are currently holding the page in Write mode, P2 is the owner
   e. P2 and P4 are holding the page in Read mode, and P4 is the owner
   f. P3 and P4 are both holding the page in write mode
   g. P1 only is holding the page in write mode
   h. P3 only is holding the page in write mode
   i. P4 and P5 are each holding the page in a Read mode, and P4 is the owner

9. Your last interview is at Facebook/Meta. Your friend who works at Facebook, has in her free time, built a new distributed file system called "FaceFS", which uses full file serving+caching like AFS but instead of AFS' "callback promise", FaceFS instead uses NFS' client caching (with timestamps) for the whole file – the latter is intended to allow two or more clients to simultaneously access a file.
   a. Give a key advantage and key disadvantage of this design decision of FaceFS.
   b. FaceFS uses *prefetching* – when a client requests a file F in a directory D, the client proactively fetches not just the entire file F, but also all the files in directory D (but not in recursive sub-directories of D). Give a key advantage and a key disadvantage of this prefetching design decision.
   c. In its client caching variant, instead of using the NFS rule for validity of the file (each time it is accessed):

      (Old NFS rule) A cache entry at time T is valid if either: 1) $(T\text{-}Tc < t)$ *or* 2) if #1 is false, then $(Tm_{client} = Tm_{server})$

      FaceFS instead uses (here t is freshness interval):

      (New FaceFS rule) A cache entry at time T is valid if either: 1) $(Tm_{client} = Tm_{server})$ *or* 2) if #1 is false, then $(T\text{-}Tc < t)$.

      Is this change a good idea? If yes, say why. If no, say why not.

   d. PS (not a question): If you're wondering why Illinois/UIUC is not listed in the above places, it's because we encourage cross-pollination and would like our alumni to spread their knowledge everywhere around the world (true fact!).

10. (For this question you can use the Web as a resource) You may have noticed that most of the classical results in distributed systems are co-authored largely by men. This is unfortunate. A welcome exception was Nancy Lynch (the L in FLP). Find at least one other woman researcher (apart from Nancy Lynch, or anyone else discussed in class) who made prominent contributions to the area of distributed systems (either industry or research). Write briefly (100 words or less) what their contribution was. The person you pick must be well-known in the field. You can provide references in your response (references do not count in the 100 word limit).

====== END OF HOMEWORK 4, and…

BEGINNING OF YOUR DISTRIBUTED SYSTEMS CAREER! =====