CS 425/ECE 428 Fall 2020 – Midterm

On-campus exam Date: October 10th (8 am US Central) – 13th (8 am US Central), 2020.

Exam time For All Students: 90 minutes. Exam will be available for 6 hours once you start. Applies to both on-campus and Coursera.

Instructions:

1. Please hand in solutions that are typed (you may use your favorite word processor to start a new solutions doc, like you do with homeworks). We will not accept handwritten solutions. Figures and equations (if any) may be drawn by hand (and scanned/photographed clearly. Please ensure what you submit is clear.

2. Please submit PDF only! Please submit on Gradescope. Do not submit on Coursera.

3. Please start each problem (Q1-Q5) on a fresh page, and type your name at the top of each page (parts of each question, e.g., Q2’s a and b, don’t need to be on fresh pages). Don’t forget to mark these pages on Gradescope.

4. This exam contains FIVE questions Q1-Q5 (pages 1-7). Please answer ALL parts of ALL questions (no choice).

5. The exam is open-book (course material) and not proctored.

6. Calculators ok. No cellphones or other electronic devices. No web access (apart from Course websites and Piazza).

7. Piazza during the above exam period (Oct 11 8 am US Central – Oct 13 8 am US Central):
   a. Please watch the running post (continuously updated) with midterm clarifications.
   b. On Piazza, you can only ask private questions to instructors, only about exam questions. Please do not email questions to the staff mailing list.
   c. There is a Piazza blackout on asking any questions about the course material, HWs, or MPs. During this time, please do not post anything visible to entire class, or post anything (visible to class, or not) about course material.

8. What you CAN use: all course material provided by instructor this semester, i.e., slides, videos, textbook, HWs+solutions, etc.

9. What you CANNOT use: Cannot use the Web, cannot communicate with other students (during exam, or afterwards, until we release solutions). You won’t gain any advantage if you do these and will only waste your precious time.

10. You CANNOT communicate with other students during the exam, or afterwards. (Note this is different from Homeworks, where you were allowed to collaborate.)

11. Answer all questions (no choice).

12. Before submitting your final doc, please make sure each question starts on a fresh page (this helps us in Gradescope to not miss your solution!).
1. Fill In the Blanks [10 X 3 = 30 points]

There are 10 questions below. For each question, write the answer, i.e., best phrase that best fills in the blanks (no justification needed). Please keep your answer concise and to the point (i.e., don’t write an essay). No partial credit.

(a) According to the CAP theorem, when a system satisfies Availability, it CANNOT satisfy ______________________

(b) A public cloud provider charges $0.05 per GB-month for storage. For a startup, the cost of buying an off the shelf storage rack of 500 TB is $200K. Focusing only on pricing, the breakeven time for preferring public cloud vs. buying+owning is __________

(c) In a gossip-style failure detection protocol, a process p whose local time is 44 has an entry for another process q that reads as (address, heartbeat counter, local time) = (q, 33, 22). Another process r contains the entry (q, 22, 33), which it sends across to process p, and p receives it at its local time 44. Process p should update its entry for q to be (____, ____, ____)

(d) A Mapreduce task is running on a cluster with 4 racks, each rack with 3 machines. Machines are numbered as S<rack number><machine number>: S11, S12, S13, S21, and so on. A Map task needs to access as input a block that has replicas on machines S22, S42, S41. The only free containers in the cluster are available at the following machines: S11, S13, S33, S43. The task will be scheduled at server __________

(e) Is the following statement TRUE or FALSE?
If two events e1 and e2 are such that e2’s Lamport timestamp is strictly higher than e1’s Lamport timestamp, then e2’s vector timestamp is also guaranteed to be strictly higher than e1’s vector timestamp.

(f) In a system of four processes sending unicasts, an event e1 has a vector timestamp of (50, 51, 52, 53) while an event e2 has a vector timestamp of (57, 56,
55, 53). The causality relation between e1 and e2 (i.e., happens before) is __________

(g) A synchronous consensus algorithm, of the kind discussed in class, is configured with 6 rounds. This algorithm can tolerate __________ number of maximum failures in a synchronous system.

(h) A version of Gnutella has a rule that “drops duplicate messages (seen before) with the same payload descriptor and Descriptor ID”. Among the four main message types (Query, QueryHit, Ping, Pong), the message type(s) that this rule should NOT be applied to is/are: __________

(i) In the heartbeat protocol, if one desired to decrease false positive rate and increase detection time, but not affect stable-state heartbeat bandwidth, the sole parameter to increase would be __________

(j) In the Grid infrastructure, the entity responsible for external allocation and scheduling among sites is __________

(Go to next page)
2. Lamport Timestamps [20 + 5 = 25 points]

This question has two parts.

(a) You’re on a date (Congratulations! Cool! Don’t mess it up.). Your date starts talking about the interactions among your 4 mutual friends—let’s call them P0 through P3. Since you’re a distributed systems person you draw out the timeline of these interactions and it looks like the figure below. Events include message sends and receipts (M*), and local steps (S*). Assign Lamport timestamps to each event. The initial timestamp of each process is 0. You can either list each event against its timestamp, or annotate figure. (If you can’t annotate, please include event names against timestamps! For messages, Mij use event names: Send(Mij) and Rcv(Mij).)

(b) Name one pair of events that are concurrent and have the same Lamport timestamps, and a second pair of events that are concurrent and have different Lamport timestamps. To name events, use (Process id, timestamp) pair.

(Go to next page)
3. P2P Routing Chips [25 points]

This question has three parts.

On your second date (congratulations!) your date challenges you to design a new P2P system. You come up with an idea to design a Chord-Kelips hybrid called Chips that works as follows. In Chips, processes are organized in a virtual ring with $2^m$ points, just like in Chord, with nodes hashed to the ring. However, Chips “splits” the ring into $K$ equi-sized number of segments, $K$ being a constant. The processes inside a segment (O(N/K) in number, on average) know everyone else in that segment (like in an affinity group). However, like Chord, in Chips file meta-information is not replicated within the segment (unlike Kelips). Unlike Kelips however (where every process has a contact in each other affinity group), Chips processes maintain a routing table across segments, with successors and finger table entries similar to Chord’s. Each process inside a segment knows a successor in the next segment (successor may be different for different nodes in a segment), and the $i$-th finger table entry ($i$ from 0 to $m-1$) at a node with id $P$ is some one process in the segment that contains $(n+2^i) \mod 2^m$

a. How would routing work in this new system? Give the routing algorithm (figures, or pseudocode, or a clear concise description). You can use an example, but we need the actual routing algorithm.

b. Analyze the per-process memory (O()), and the latency (O()) of routing (informal but clear justification is ok, formal proof not needed).

c. When failures occur, is this variant more fault-tolerant than Chord or less? Say clearly why? (give the alternate routing paths).

(Go to next page)
4. MapReduce [20 points]

Now that you’re back from your date, you really need to get to work. Your boss gives you a dataset from a symmetric social network (like Facebook) where a is a friend of b implies that b is also a friend of a. The input is a file (sharded) containing such pairs (a, b) - note that either a or b may be a lexicographically lower name. Each line of input is one such entry (input to initial Map function), and lines are not duplicated (i.e., no (a,b) and (b,a)). The intern wants to answer the following question: “Among the set S of all users with first name ‘Joe’, what fraction (or percentage) of such users (in S) are such that they are friends with at least one user with first name ‘Donald’?” You can chain Mapreduces if you want (but only if you must, and even then, only the least number). You don’t need to write code – pseudocode is fine as long as it is understandable. Your pseudocode may assume the presence of appropriate library primitives (e.g., “getfirstname(user_id), etc.). The Map function takes as input a tuple (a,b).

You should have at least some parallelism. You should try to minimize the total amount of data that is shuffled (from Map to Reduce) and sent to/from HDFS (from Reduce to the next Map). Ensure that your output does not contain duplicates. You can set your key and value to arbitrary objects. You cannot retain data at any of the machines from a task (Map or Reduce) for use in a later task. Chaining MapReduces is allowed, as long as you don’t over-use chaining (where parallelization could have helped instead). Each MapReduce in a chain can read the dataset, but there is no other persistent memory across the chain (apart from the outputs of stages). Please write clear pseudocode, and is preferable. Be clear and concise and don’t miss any steps.

(Go to next page)
5. Failure Detection [15+5 = 20 points]

This question has two parts.

You’re on another date (lucky you!), and you start complaining about how hard HW1 in CS425 was. (Tip: Don’t complain about your own life on a date.) Anyway, your date asks to see your HW1, and designs a failure detection protocol, for an asynchronous distributed system, that is a modified ring failure detection protocol. It works as follows: there are $N$ processes, each process $i$ selects a heartbeat set $HB(i)$ consisting of:

i) process $i$’s $k$ successors and $k$ predecessors in the ring ($k \ll N$), as well as ii) $k$ random processes that are not amongst process $i$’s $k$ successors/predecessors. Once $HB(i)$ is selected, it doesn’t change, i.e., these are selected at the protocol start and are not changed thereafter. All processes in $HB(i)$ send heartbeats to process $i$. Heartbeats are not relayed, but instead recipients do the same action as processes in the ring heartbeat protocol recipients discussed in class, i.e., process $i$ times out if it doesn’t receive heartbeats from process $j$ within a timeout and marks $j$ as failed. A process is detected as failed if any one of its non-faulty heartbeat receivers do not receive expected heartbeats within a pre-specified timeout.

(a) What is the max number of failures this algorithm can tolerate before it violates completeness? (If your answer is M, then any failures up to M in count do not violate completeness, but some scenario with M+1 failures does violate completeness.) Give a proof or counter-example, don’t just give a count.

(b) The intern also claims this algorithm satisfies accuracy. Are they right? Argue why or why not (informally, but clearly proof).

(END OF EXAM)