15-440/640 Distributed Systems
Midterm

Name:	
Andrew ID:	

October 18, 2018

- Please write your name and Andrew ID above before starting this exam.
- This exam has 21 pages, including this title page. Please confirm that all pages are present.
- This exam has a total of 100 points.

Question	Points	Score
1	15	
2	20	
3	9	
4	12	
5	10	
6	16	
7	16	
8	2	
Total:	100	

True/False

1. Circle whether each statement below is *true* or *false*. ***ALSO***, give a one to two sentence reason for your answer.

   Each correct answer is worth 3 points (1pt for the T/F and 2pts for the reason).

   (a) (3 points) [True, False] CODA uses pessimistic replication to ensure high-availability to users.

   (b) (3 points) [True, False] Write Ahead Logging for transactions is used to ensure both Atomicity and Isolation properties of ACID.

   (c) (3 points) [True, False] In the context of data stores, having eventual consistency does not imply that the data store is also sequentially consistent.

   (d) (3 points) [True, False] Kerberos is a system used to throttle users who are abusing their access to a DFS (e.g., making too many requests, consuming too much bandwidth).

   (e) (3 points) [True, False] Under LSP protocol from P1, after the initial connection setup phase, we can optimize network bandwidth by not sending out ACK 0 messages when the epoch timer fires, as long as the network does not reorder messages. There would be no change in the behavior of LSP protocol.

Short Answers

2. In the following, keep your answers brief and to the point (i.e. 2-3 sentences).

   (a) (4 points) You are responsible for building the successor to the popular Skype Voice-over-IP protocol that works across the internet (high RTTs). Your boss went to CMU but did not take 441 or 440, and thinks you should use TCP due to its reliable delivery feature. Can you explain to him why this is not a good idea?

   (b) (4 points) NoSQL systems like Amazon's Dynamo allows writes to continue to all machines during a network partition. Once reconnected, Dynamo nodes seek to reconcile inconsistent file versions. Which technique would you choose to detect and flag parallel events? Briefly explain why.

   (c) (2 points) Name and briefly describe two differences between the time synchronization algorithm used in NTP and Christian's algorithm.

You want to design a file system that can tolerate up to f node crash failures. For each of the following scenarios, how many file servers do you need at least? State the values and explain in one sentence.

(d) (2 points) The file system is using Primary-backup to replicate the files.

(e) (2 points) The file servers are using Basic Paxos to elect a leader among themselves.

Consider two proposers A and B in the Paxos algorithm. Suppose that A is starting phase 2 (accept) and B is starting phase 1 (prepare). Assume that participants will not crash throughout the process.

(f) (6 points) Sketch out a scenario where the Paxos algorithm does not achieve the liveness property (a labeled sketch is fine, you can annotate a part that is "repeated forever").

Communication and RPC

3. You are hired by the augmented reality startup AR^2 that needs to perform a complex image processing algorithm. Because mobile phones have little computing power, AR^2's algorithm gets unacceptably slow as image sizes increase. Your manager, Jaine, proposes to use RPC to perform the work on a remote, more powerful server.

   You measure the following numbers, assuming a base image scaling factor of N.

   - The running time of AR^2's algorithm is $M(N) = 3N\ ms$ on a mobile phone. On the server, its $S(N) = N/3\ ms$.
   - Marshalling and unmarshalling take $(N/6 + 20)\ ms$ in total.
   - The average RTT between a user and the server is 80 ms.
   - Bandwidth is infinite between the user and the server.

   (a) (4 points) For what value of N is Jaine right, i.e., AR^2 should be using an RPC instead of a local call?

   (b) (2 points) With geo-replication, AR^2 is able to reduce the average RTT. How will N change? (Give a qualitative answer)

   (c) (3 points) As AR^2 acquires 1) more and more powerful servers and 2) places them closer to users, you notice that the total RPC time does not improve very much. Explain why, by pointing out the bottleneck, in two sentences or less.

Distributed File Systems: since really everything that was innovative in DFS's was invented @ CMU !!

4. Karan has decided to drop out of CMU, work with Peter Thiel since he believes that education is overrated. He wants to create the next great cloud file storage company called "SoftBox". However, his starting funding is very low, since no one wants to give money to him. This means his Softbox system has occasional network partitions due to node failures, and high network latency.

   (a) (3 points) Karan wants to market that he can provide two guarantees: (1) That all file changes will be *immediately* visible to all other clients. (2) Any client can access any file at any time. Explain whether or not this is possible? (Hint: Recall that his funding is very low.)

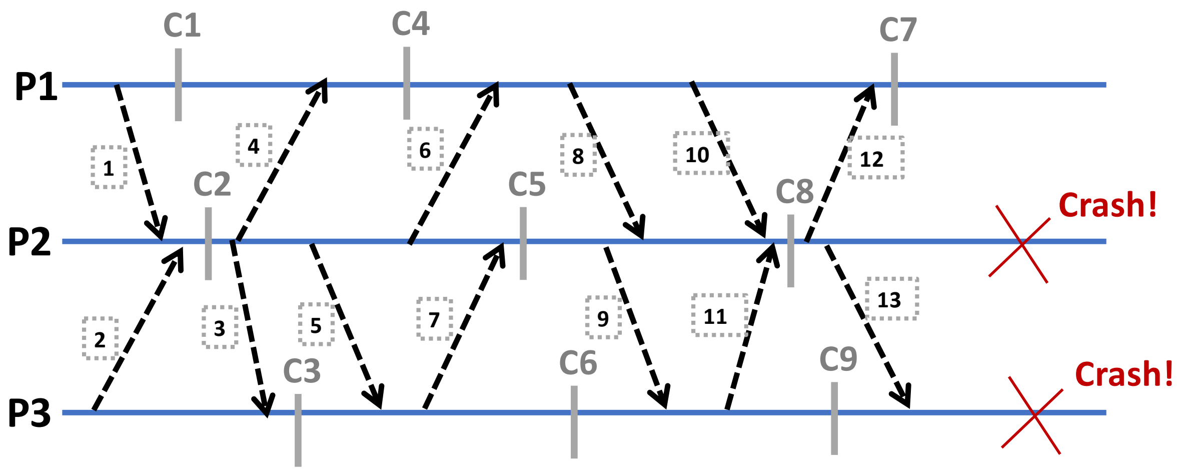
   (b) (3 points) Karan argues that companies like Dropbox and Google Drive are successful because they provide the same guarantees as he is aiming for in Softbox. Is Karan right? Briefly explain your answer.

   (c) (3 points) Karan decides to relax his requirements a bit to get to market faster. He decides to prioritize "Any client can access any file at any time". Should Karan implement his system with optimistic or pessimistic replication?

   (d) (3 points) Karan is worried that his users will experience extremely high latency every time they request a file. What is a simple technique to improve user experience?

Logging and Failure Recovery

5. You have designed a new distributed system, that is highly scalable and uses 3 nodes, P1, P2, P3 to send messages to each other (shown in square gray boxes). Since you are a smart CMU student you are convinced that node failures are a reality and you must prepare for the worst and bring back your entire application to a consistent state. You like the idea of checkpointing, and each node in your system takes independent snapshots after sending or receiving at most 3 messages. (C1 - C10 in the example below). Now, the following happens:



(a) (5 points) Process P2 and P3 both fail. Can your system recover from this failure and get back to a consistent state using the checkpoints? If so, calculate a consistent recovery line (listing the snapshots) for rollback. Note:

1. If there is no recovery line state why.
2. If there are multiple, list the ideal recovery line. Explain your choice.

(b) (5 points) Next, you realize that both checkpoints C5 and C8 taken by P2 were actually corrupt on the disk. Can your system still recover to a consistent state by using the other stored checkpoints? If so, explain how. If not, explain why not, and give one efficient mechanism on how you could address this issue going forward.

BergerNet

6. BergerNet is a new social network started by the TAs and Professors of 15440 providing almost the same functionality as social networks do today. As a new engineering lead, it is your job to make some crucial design decisions that are either going to make or break the BergerNet.

   BergerNet uses RPCs. Identify the easiest-to-implement RPC semantic (at-least-once, at-most-once, exactly-once) for each of the following features. Give a *one sentence* reason for your choice. (1pt for semantic, 1pt for reason only if semantic correct).

   (a) (2 points) Transferring money to your friend on BergerNet

   (b) (2 points) Posting a message in the BergerNet messenger.

   (c) (2 points) Deleting a friend on BergerNet

Next, BergerNet users want to upload group collages of their friends and colleagues. Privacy is important. So before a user can upload and publish a collage, the user needs to get approval from all their friends and can only upload the collage if all of them approve. Even if one friend rejects the proposal, the collage cannot be uploaded.

(d) (1 point) Name a protocol from lectures, with the least number of messages, to implement the collage feature.

(e) (2 points) Describe each stage of this protocol with a bullet point in the given context.

(f) (1 point) If the computer of the user who uploads fails during this protocol what could be one negative outcome other than the upload failing?

(g) (2 points) Is there a way of solving this problem? If yes, name a protocol that resolves the issue, if no, explain in one sentence why.

BergerNet is very popular these days mainly because of how little it crashes.

(h) (2 points) Define reliability and availability (one sentence each).

- Reliability

- Availability

Let's take the previous year as an example. BergerNet crashed once every hour and recovers within a second.

(i) (2 points) Would you say that BergerNet is highly available and/or highly reliable or neither ?

Hepp Dean's Hierarchical Mutual Exclusion

7. In 2027, Foodle is the world's leading Internet company with ten datacenters around the globe. This future version of the Internet has neither packet loss nor network partition nor sudden machine crashes, but the speed of light remains as a major constraint. A critical piece of Foodle's algorithms require mutual exclusion across all servers in all data centers.

   Foodle's design head, Hepp Dean, proposes a new hierarchical distributed mutex. Each datacenter has a unique coordinator, `M`. Across the ten datacenters, the ten coordinators uses Ricart & Agrawala's algorithm, which has the following thread-safe API: `BroadcastRequest()`, `GotGlobalLock()`, `BroadcastRelease()`.

   Within each datacenter, the coordinator manages datacenter-local mutual exclusion among the local client machines. Below is Hepp's pseudo code for coordinators.

```
M.Init():
# Initialize this leader machine
  M.hasRequestedGlobalLock = False
  M.hasGlobalLock = False
  M.Queue = NewQueue()
  M.mutex = 0
  M.cvar = NewCond(M.mutex)

M.GotGlobalLock():
# Triggered when M gets mutual exclusion among other leaders
  M.mutex.lock()
  M.hasRequestedGlobalLock = False
  M.hasGlobalLock = True
  M.cvar.SignalAll()
  M.mutex.unlock()

M.Lock(id):
# Client with id calls this function using RPC when asking for mutual exclusion.
  M.mutex.lock()
  if not M.hasRequestedGlobalLock && not M.hasGlobalLock:
      BroadcastRequest() // Broadcast to other leaders for mutual exclusion
      M.hasRequestedGlobalLock = True

  M.Queue.PushBack(id)

  while not M.hasGlobalLock || M.Queue.Front() != id:
      M.cvar.Wait()

  M.mutex.unlock()

```

```
M.Unlock(id):
# Client with id calls this function through RPC when releasing mutual exclusion
  M.mutex.lock()
  M.cvar.SignalAll()
  M.Queue.Pop()

  if M.Queue.Empty():
    M.hasGlobalLock = False
    M.BroadcastRelease() //Broadcast to other leaders to release mutual exclusion
  M.mutex.unlock()
```

(a) (4 points) Across different datacenters and in the absence of failures, explain briefly why Hepp's algorithm safely implements a mutex?

(b) (4 points) Within a single datacenter and in the absence of failures, explain in less than three sentences why Hepp's algorithm safely implements a mutex?

(c) (2 points) Assume that many client machines simultaneously seek to acquire the mutex. In what order will they acquire it? Describe the order for both cases below.
All client machines in the same datacenter:

Client machines in different datacenters:

(d) (3 points) State an advantage of Hepp's hierarchical distributed mutex, assuming that there are tens of thousands of servers in each datacenter?

(e) (3 points) State a disadvantage of Hepp's hierarchical distributed mutex, in the context of the mutex design goals discussed in class? How would you resolve it?

Anonymous Feedback

8. (2 points) Tear this sheet off to **receive points**. We'd love it if you handed it in either at the end of the exam or, if time is lacking, to the course secretary.
   (a) Please list one thing you'd like to see improved in this class in the current or a future version.

   (b) Please list one good thing you'd like to make sure continues in the current or future versions of the class.