2. (5 points) What are the design issues of distributed scheduling?

Issues	Affects
Communication, Synchronization, distributed algorithms	Interaction and Control
Process scheduling, deadlock handling, load balancing	Performance
Resource scheduling, file sharing, concurrency control	Resource
Failure handling, configuration, redundancy	System Failures

3. (5 points) Why is message logging useful in Checkpointing? Discuss.

There's no such a system with zero defects even it is developed by a professional development team. Developers write code and they need to find problems in their applications during and after development, however, some problems can only be founded during runtime. Message Logging are often deployed in places where they don't have access or there are no debugging tools however, it makes it possible to help developers to locate problems. For instance, in game hack development, we usually inject a DLL into system process, however, system processes are run in another session, in where we cannot spawn a debug window to output error messages / debug info. Thus, message logging using file are suitable for this case and can help to locate bugs / errors. Developers can now exactly what program did by looking at message logging. As well as MySQL general log, which is message logging and very useful for backend development.

Another thing is, checkpointing is expensive which requires Incremental snapshot. However, message logging is cheap, combine with message logging with checkpointing, the system has ability to build single replay, Piecewise deterministic model, as well as Orphan process (crash before store, in inconsistent state after recovery)

4. (5 points) What is Byzantine failure? How many replicas are needed to survive a k component fail in Byzantine failure?

A typical Byzantine fault is defined as a fault presenting different symptoms to different observers, for example, arbitrary deviations of a process from its assumed behavior based on the algorithm it is supposed to be running and the inputs it receives. A Byzantine failure is the loss of a system service due to a Byzantine fault in systems that require consensus.

If there are K component fail, in order to survive, all servers must receive the same messages, and execute commands, thus in typical design, K replicas are needed to survive in this failure.

5. (5 points) In the Bully algorithm, a recovering process starts an election and will become the new coordinator if it has a higher identifier than the current incumbent. Is this a necessary feature of the algorithm? Explain.

Actually, this feature of algorithm is undesirable in case there is no advantage to using a higher-numbered process: the re-election is wasteful.

However, for example, with higher-numbered processes executing at faster machines, the numbering of processes may reflect their relative advantage. In this case, the advantage may be worth the re-election costs. Re-election costs include the message rounds needed to implement the election; they also may include application-specific state transfer from the old coordinator to the new coordinator. A recovering process could merely send a request Status message to successive lower-numbered processes to discover whether another process is already elected, and elect itself only if it receives a negative response to avoid re-elect. Thereafter, the algorithm can operate as before: if the newly-recovered process discovers the coordinator to have failed, or if it receives an election message, it sends a coordinator message to the remaining processes

6. (5 points) Suppose that two processes detect the demise of the coordinator simultaneously and both decide to hold an election using the bull algorithm. What happens?

The election will proceed as usual.

Each of the higher-numbered processes will get two elections messages, but will ignore the second one.

7. (5 points) Explain why there are two campus of file systems, represented by Parallel File Systems and Hadoop File Systems?

The long-standing wisdom is that RAID is not beneficial for Hadoop data nodes.

Distributed file system such as Hadoop file system has redundancy anyway, and RAID-0 slows down the entire array to match the speed of the slowest drive in the array. A parallel file system aggregates the bandwidth from many storage nodes to feed a compute node. While Hadoop was able to achieve its performance through its clever insight of shipping code to data, each CPU in a Hadoop cluster has to suck its data from a single disk through a straw.

8. (10 points) What kind of consistency would you use to implement an electronic stock market? Explain your answer.

I would use Causal consistency to implement.

This can handle the issue that reactions to changes in stock values should be consistent. Changes in stocks that are independent can be seen in different orders.

9. (10 points) Consider a personal mailbox for a mobile user, implemented as part of a wide-area distributed database. What kind of client-centric consistency would be most appropriate? Explain.

The issue is very clear, that the owner should always see the same mailbox, no matter whether he is reading or updating it. In fact, the simplest implementation for such a mailbox may well be that of a primary-based local-write protocol, where the primary is always located on the user's mobile computer.

10. (10 points) Give an example where client-centric consistency can easily lead to write-write conflicts.

Suppose that each data item had a single owner, which had exclusive write access in client-centric consistency models. In case of two independent users of the same data eventually bind to the same replica, their respective updates may need to be propagated to that replica. In this situation, there would be write-write conflicts.

11. (10 points) A file is replicated on 10 servers. List all the combinations of read quorum and write quorum that are permitted by the voting algorithm.

The following possibilities of combinations of read quorum and write quorum are permitted.

(1, 10), (2, 9), (3,8), (4, 7), (5, 6), (6, 5), (7, 4), (8, 3), (9, 2), (10, 1).

12. (10 points) Explain the difference between linearizability and sequential consistency, and why the latter is more practical to implement, in general.

Both care about giving an illusion of a single copy. From the outside observer, the system should behave as if there's only a single copy, however, Linearizability cares about time but Sequential consistency cares about program order. With sequential consistency, the system has freedom as to how to interleave operations coming from different clients, as long as the ordering from each client is preserved. So, the latte is more practical to implement.

13. (10 points) Consider a Web browser that returns an outdated cached page instead of more recent one that had been updated at the server. Is this a failure, and if so, what kind of failure?

Either response failure or performance failure. It depends on the policy that browser provide content to user. If the browser designed to provide pages that are at most T time units old, it may exhibit performance failures. However, a browser can never live up to such a promise in the Internet. Simply returning a page from the cache without checking its consistency may lead to a response failure.

14. (10 points) In reliable multicasting, is it always necessary that the communication layer keeps a copy of a message for retransmission purposes?

No, it's not necessary. In many cases, it is necessary only that the data is still available at the application level thus therefore maintain a copy for retransmission. There is no need that the communication layer maintains its own copy.

15. (10 points) In the two-phase commit protocol, why can blocking never be completely eliminated, even when the participants elect a new coordinator?

For example, in case of the new coordinator crash as well after the election, the remaining participants cannot obtain a final decision, because this requires the vote from the newly elected coordinator, just as before.

16. (10 points) The totally-ordered multicasting using Lamport's logical clocks does not scale. Explain why.

The messages can be lost, messages from the same sender are received in the order they were Sent can changed. So it does not scale.

17. (10 points) Devise a simple authentication protocol using signatures in a public-key cryptosystem.

If Alice wants to authenticate Bob, she sends Bob a challenge R. Bob will be requested to return KB(R), that is, place his signature under R. If Alice is confident that she has Bob's public key, decrypting the response back to R should be enough for her to know she is indeed talking to Bob.