CS582 Distributed Systems
Quiz 8

Student Name:		**Student ID:**
Time Allowed: 20 mins.	**Total Marks:** 12	**Marks Obtained:**

1. (3 marks) Danish, an ML researcher, has recently found out about *Scaling Distributed Machine Learning with Parameter Server*. He has decided to make his model's training phase more efficient using his own implementation of parameter server having one parameter server node and 10 worker nodes. His workload involves terabytes of parameters and training examples.
   Identify and briefly explain three potential bottlenecks with Danish's approach.
   Answer:

   1. **Processing delays:**
      All push and pull requests for the parameters and updates from all the workers will be sent to the same parameter server which could possibly cause there to be processing delays as the server would have to process all the requests on its own.
   2. **Single point of failure:**
      The same server is responsible for updating the parameters. If it crashes, we would lose all parameters and updates that have already been made.
   3. **Network congestion:**
      Even if worker nodes pull only the parameters relevant to their training data, all parameter updates and values must be routed to the same single parameter server. This concentration of traffic can saturate the network link to the server, leading to congestion.

2. (3 marks) Select **all** the correct statements from the statements below. You will lose marks for selecting any wrong options, but your total score for this question cannot go below zero.

   A. Using a distributed parameter server setup with X server nodes and Y worker nodes, we can achieve a maximum potential speedup of XY times over a setup with 1 parameter server and 1 worker.

   B. For fault recovery, Spark logs a single operation that applies to many elements rather than logging on a per-element basis.

   C. The Map() function in MapReduce is called on each key value pair to output an intermediate $< k2, v2 >$ pair.

   D. Each operation on a Resilient Distributed Dataset (RDD) in Spark will produce a new RDD, and both of these RDDs are stored in the RAM.

   For the other variant, the first three options are correct. The last option was changed to say disk rather than RAM and the first option was changed to a speed up of Y times.

3. (2 marks) Hammad is a big fan of the MapReduce framework and uses it for all his tasks. However, Ayain suggests that if an operation in the *Map* function is non-deterministic, he can run into inconsistencies across the final R files produced by R reduce workers.
   Do you agree or disagree with Ayain? Justify your answer.
   Answer:
   Agree.
   If a worker node fails, the map tasks given to this worker are reset back to the initial state and rescheduled for other workers. However, if the *Map* function is non-deterministic, rerunning the same task may produce different intermediate results. This difference arises because the non-deterministic *Map* function might rely on factors such as random values, timestamps, or external state.
   When reducers begin processing, some may have already consumed intermediate data produced by earlier executions of the *Map* tasks, while others may read new outputs from the rerun *Map* tasks. Consequently, this leads to inconsistencies in the inputs received by reducers, resulting in discrepancies in the final output.

4. (4 marks) MangoDB, a new Big Data company, handles data storage and processing for their customers. They currently use MapReduce for fault tolerance and parallelization of their workloads. Their customer base primarily works with data mining applications where users can run queries interactively, and run many queries on the same data.
   Unfortunately, their customers have been complaining about slow processing speeds and their ratings are going down. MangoDB holds an internal meeting to address this. One of the team members, having taken the Distributed Systems

course at LUMS, says that MangoDB should consider switching to Spark instead of using MapReduce. But other people in the MangoDB are skeptical about this choice. Your job is to help MangoDB pick the right option.
(2 Marks) Is MapReduce a good option for MangoDB? Justify/explain your answer.
Answer:
No. In MapReduce, the only way to share data across jobs is through persistent storage. For workloads like data mining, where many queries are run on the same data and queries are also run interactively, MapReduce suffers from repititive reads and writes to persistent storage. Since disk I/O is very slow, MapReduce is a bad option in this case.

(2 Marks) Is Spark a good option for MangoDB? Justify/explain your answer.
Answer:
Yes. Spark uses RAM instead of persistent storage for data processing so it is much faster than MapReduce. For data mining, where users run interactive queries on the same data, Spark would not incur the overhead of slow disk reads for each query since data would be available for use in the RAM. Therefore, it would be much faster than MapReduce and switching to Spark would be a good option for MangoDB, significantly improving the processing speeds for their customers.