Willis Allstead
11/8/18
CS 491

Review: The Hadoop Distributed File System

This paper is very concise, but it does a good job at explaining all of the critical concepts of the Hadoop Distributed File System (HDFS) clearly. The authors waste no time explaining the exact paradigm Hadoop is built using, MapReduce. They also write early in the paper about how widely-used Hadoop was at the time of writing. One problem with how the presented the paper was that they didn't clearly state the problem they were trying to solve, besides a quick statement in the abstract. Essentially HDFS was created to store very large datasets reliably, and to stream those datasets with high bandwidth. This problem is significant for many reasons because, depending on the company, they might need to push significant requests for data through their system. They might need to show thousands of results at once, like in the case of Yahoo, or they might need to store massive amounts of data, like Netflix. This problem is hard because to achieve both large data sets and be able to stream them with high bandwidth, you usually have to sacrifice reliability and security.

Before Hadoop and HDFS came along, the state-of-the-art technologies were PVFS, Lustre, and GFS. Compared to these solutions, HDFS is a faster and more efficient filesystem implementation. This is done, in part, by HDFS by not using data protection mechanisms like RAID. HDFS follows in GFS's footsteps by ensuring data durability by replicating file contents on multiple DataNodes for reliability. HDFS also achieves higher bandwidth by employing techniques like keeping the entire namespace for the filesystem in RAM.

Another technique that HDFS uses to improve performance is providing an API that exposes the locations of a file blocks in the file system. This enables MapReduce, a significant computation component of Hadoop, to schedule a task to where the data are actually located, rather than having to search for the file each time a task must run. This API also exposes the ability to customize the replication factor for a file, which by default is 3. The paper neglects to compare HDFS to its alternatives when it comes to performance, opting to instead simply run average Read/Writes per second per node and operation throughput per second.

The authors used these metrics of average Read/Writes per second per node and operation throughput per second to convince the reader that HDFS is superior to the alternatives of the time. They used a benchmark called DFSIO to measure the average read and write operations. DFSIO is included within Hadoop and is built specifically to measure its performance. The way it works is by using MapReduce to read/write random

data from/to large files, and records the amount of time each task takes. This measurement is for performance of data transfer only, and doesn't include the time for task scheduling, startup, and the reduce task. They found that the Read was 66 MB/s per node and the Write was 40 MB/s per node, which makes sense since writes always take time to add data to a store.

These metrics seem great, but again the authors fail to give us a reference point to compare the values to. On top of that, they use their own self-built benchmarking algorithm which really means that they are only benchmarking HDFS's performance against its own prior performance. They really should have either created a standard benchmark to test a number of other filesystems against and reported the results, or just found one out there already if it existed by then.

One key issue with HDFS and Hadoop in general is that a Hadoop cluster is completely unavailable if a NameNode is down for any reason. They mention in the paper that programmers at Yahoo were working on a solution for this at the time of writing, so it could be fixed by now, but an automated failover solution arguably should have been included in one of the earlier versions of HDFS.

Overall, this paper was well-written and does a good job at touching all of the most important ideas of HDFS. The authors could have done better jobs at providing comparisons to other filesystem implementations, but they were obviously focusing their efforts in the paper on explaining the inner workings of HDFS. Possibly if Hadoop didn't have such a large business backing it as Yahoo, they would have had more of an incentive to convince the reader to try their filesystem.