Machine Programming 3 – Simple Distributed File System

Yunsheng Wei (wei29) Neha Chaube (nchaub2)

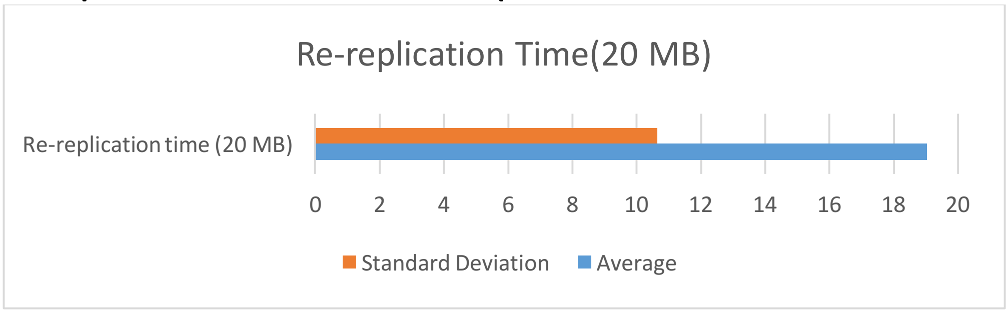
In MP3, we developed a Simple Distributed File System (SDFS). Some key design decisions are as follows:

1. SDFS is based on master slave architecture. There is one master node (name node) and multiple slave nodes (data nodes). SDFS is built on top of group membership service and leader election service. When the name node dies, the leader election service will elect a new name node among alive nodes.
2. Name node stores meta data about the whole SDFS (e.g., which file is stored on which data nodes, etc.) When name node detects the failure of a data node, it will delete from meta data information about that data node. Name node periodically checks whether there is some file needed to be replicated, and if any, it will send replication request to corresponding data nodes. Once put a file on a data node, the data node will inform the name node, so name node can update the meta data.
3. When a client performs operations to SDFS, it will first contact name node. For "get" command, name node will reply with a list of data nodes, and client will fetch file from one of the data nodes in order until success. For "delete" command, name node will then directly send delete command to corresponding data nodes. For "put" command, name node will reply with K idlest data nodes, and client will send put command to all the K data nodes. (K is the replication factor)
4. When a new name node is elected, all data nodes will send block report (containing information about SDFS files present on that data node) to the newly elected name node, so that the new name node will rebuild the meta data for SDFS. When a name node first starts, it will have a Silent Period, during which, it will not perform replication checking in order to avoid bogus replication.
5. All communication among data nodes, name nodes, and clients are using Java RMI. Leader election service and group membership service are designed using Observer Pattern.

We fixed some potential race conditions for our design, and also got the results for our experiments by using MP1's distributed log querier.
The leader election service is built on top of MP2's group membership service. And name node service is also built on top of MP2's group membership service.

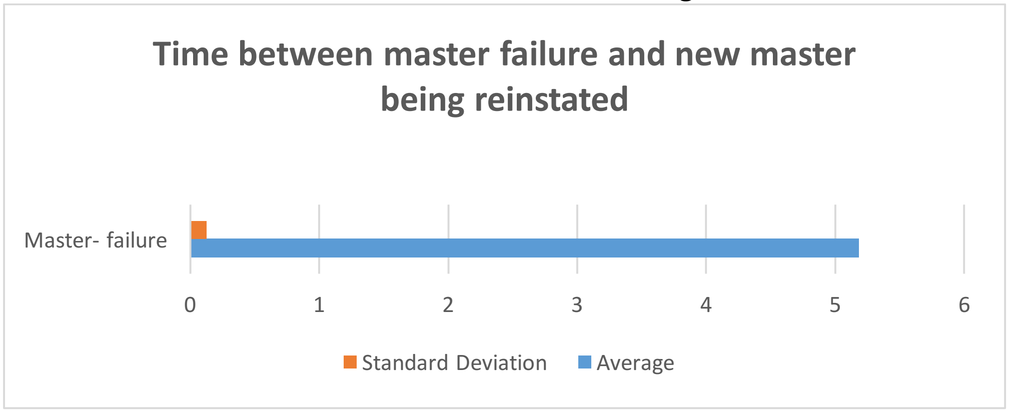
Experiments

i. **Re-replication time and bandwidth upon a failure**



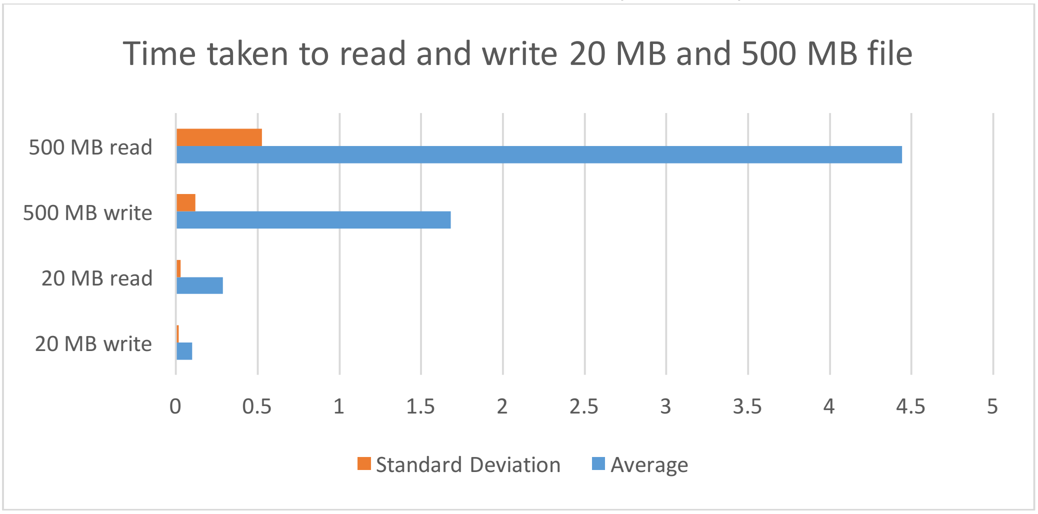
The average time taken to replicate 20 MB file upon a failure is 19.041 seconds and the standard deviation is 10.643 seconds. Because in our design, the name node periodically checks whether there is some file needed to be replicated (we set the period to be 20 seconds), so the worst case can be that the name node just finishes the last round of replication checking, and the best case can be that the name node just about to start this round of replication checking. We observe the bandwidth usage to have an outburst of roughly 20MB in a short time when the file is being replication, but this is hard to measure, so we do not plot it.

ii. Time between master failure and new master being reinstated



The average time taken between master failure and new master being reinstated is 5.181 seconds and the standard deviation is 0.125 seconds.

iii. Times to read and write one file of size 20 MB, 500 MB, under no failure

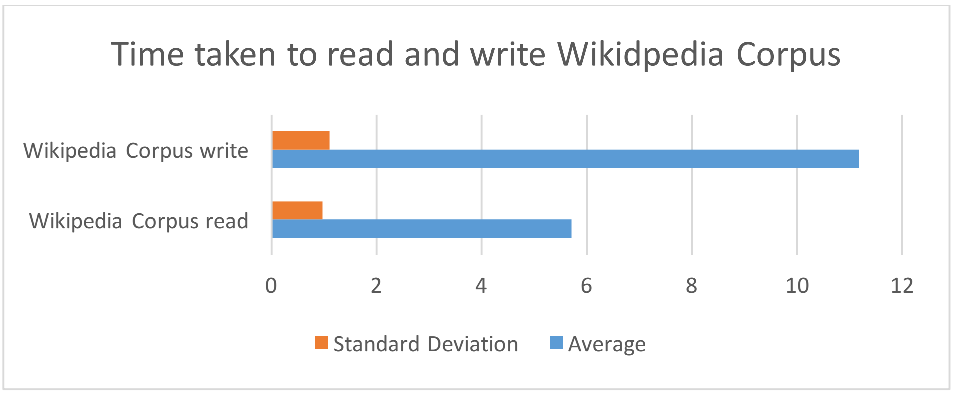


The read operation is faster than write operation. It is as expected, because read operation only needs to read from any one data node, but write operation requires client to put file on all 3 data nodes. (3 is the replication factor) The actual ratio of time for operation on 500 MB and 20 MB file is less than 25, it is as expected, because it takes extra time to set up connection.

iv. Time to store the entire Wikipedia corpus into SDFS with 4 machines

The average time taken to store 1.4 GB Wikipedia Corpus is the maximum and it takes almost 6 seconds more and the standard deviation is increased by 0.8 as compared to 20 MB and 500 MB files.

The average time taken to read 1.4 GB Wikipedia Corpus is the maximum and it takes almost 4 seconds more and the standard deviation is increased by 0.4.


Time taken to read and write Wikidpedia Corpus
Wikipedia Corpus write
Wikipedia Corpus read
0
2
4
6
8
10
12
Standard Deviation
Average