CS-425 MP2 Report   Group No: 44

Lin Lyu(linlyu2), Ching-Hua Yu(cyu17)

I. Design

Master Election: machines in a ring maintain a master consensus by attaching the master is into the heartbeat. Once a node failed and it is master, machines will elect the node with the smallest id in the ring to be the new master and this message will be attached to heartbeat again to ensure consensus.

Replication Strategy: Every file has 4 replicas in the SDFS to service less or equal to 3 failures. Once a node failed, if it is not the master, master will notice its failure according to the change of membership list. Master will lookup all replicas in this node. For each replica, master will select one node(node_a) in the membership list and one node(node_b) having this replica, and ask node_b send node_a this file. If master is failed, the new master will ask every alive machine to hand-in the sdfs file name list. If there are files which don't have enough replicas(because they have one in previous master), the new master will ask a node with the file to duplicate one on a node who don't have this file.

PUT: A node will send put request to master, and master will check whether the file is in the SDFS. If the file does not exist, the master will pick 4 nodes and sends id of those nodes to the node request a put. The node connects those nodes after receiving the acknowledgement from master and send the file. If the file already exists and more than 1 minute past since last put, the master will allow this node to use its replica replace original file. If the update is within 1 minute, the master will ask the machine to confirm its action within 30s. We use Quorum strategy to write and read. A put is regarded as success only if every replica of this file is updated.

GET: A node planning to read a file in SDFS will first ask master. The master will return a node id who has replica of this file if the file exists. After that, the node will ask that node to get the file. Since we adopt W = N, the system can read any replica of a file.

DELETE: The node will ask master and master will send the node list back if the file exists. After that the node will ask those nodes to delete the file.

LS: The node will ask master and master will send the list back if the file exists.

STORE: Print out SDFS replicas one machine has.

II. How MP1 used in MP3

We logged actions to files(PUT/GET/DELETE) and node fail or join. And we can query logs from any machine to debug easily.

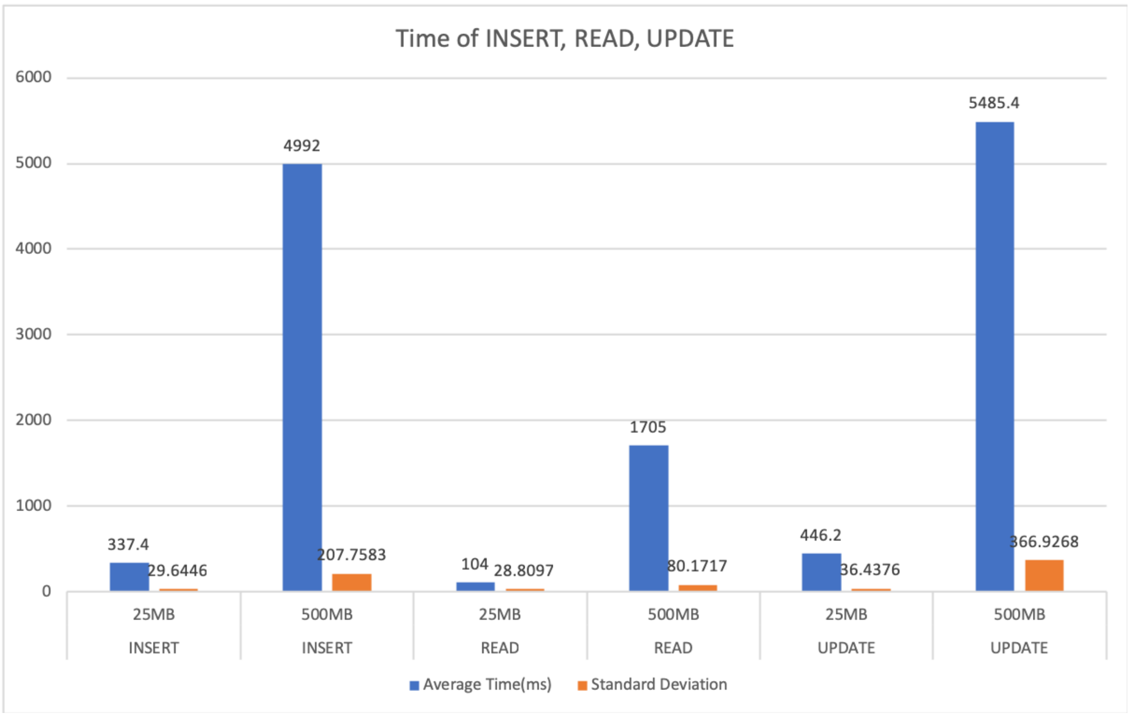
III. Measurements

1) re-replication time and bandwidth (Bps not messages per sec) upon a failure

Average time: 87ms, bandwidth: 115MBPS

2) times to insert, read, and update, file of size 25 MB, 500 MB

ACTION	INSERT		READ		UPDATE	
SIZE	25MB	500MB	25MB	500MB	25MB	500MB
Average Time(ms)	337.4	4992	104	1705	446.2	5485.4
StandardDeviation	29.6446	207.7583	28.8097	80.1717	36.4376	366.9268
1	354	4924	131	1655	439	5023
2	376	5040	113	1717	498	5892
3	341	4718	107	1801	467	5398
4	306	4985	114	1754	412	5291
5	310	5293	55	1598	415	5823



The time increases dramatically when inserting a 500MB file compared to a 25MB file. It indicates that READ is much faster than WRITE. We believe this is due to the Quorum mechanism we used, which is to wait for all replica acks when writing but only get any copy when reading. This is a great fit for a read-heavy system.

3) time to store the entire English Wikipedia corpus into SDFS

	4 machines	8 machines
Average	10719 ms	12758 ms
Standard Deviation	672.153 ms	619.587 ms

As the table shows, the average time to put WIKI corpus is similar when we have 4 or 8 machines. We think this is reasonable because the system have the same number of replicas no matter how many machines.