CS425, Distributed Systems: Fall 2017

Machine Programming 3 – Simple Distributed File System

Team Member: Haowen Jiang, Tiancheng Wu  
NetID: haowenj2, twu54

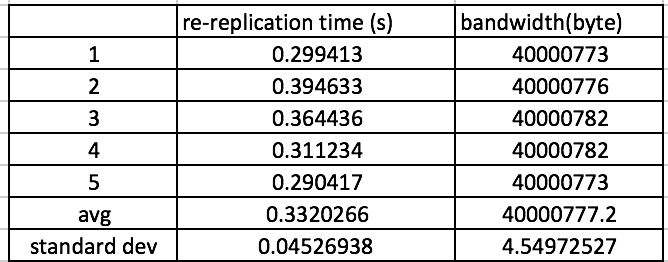
# Design:

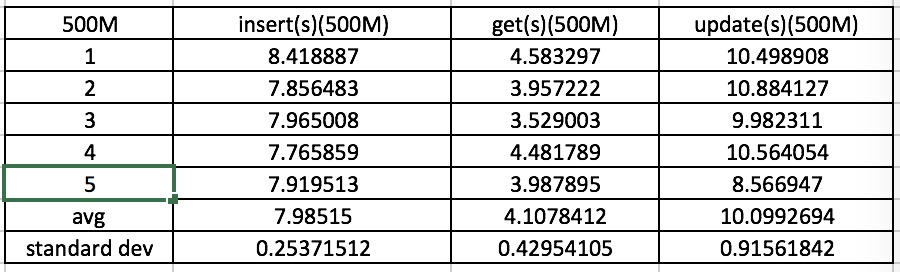
The purpose of this project is to build simple distributed file system based on Mp2 – distributed group membership. The whole project can be separate into five functions: put file, get file, delete file, list file saving directory and list file name saved in local machine.

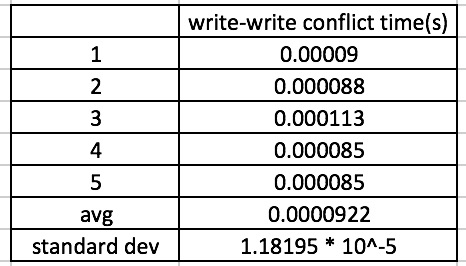
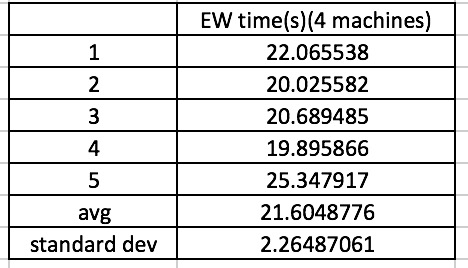
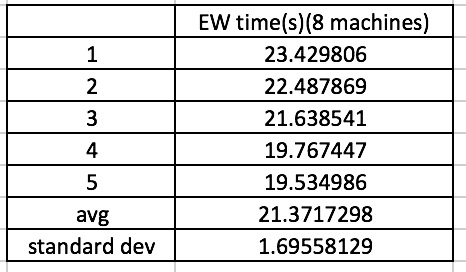
* Put-file: Since the system should tolerant 2 machines illegal exit, at least three replicas exist. Meanwhile, quorum methods should be applied in this project which means the whole system should have at least 3 up to date replicas. Put-file function should write at least 3 files every time. The idea of writing file to the other machines are based on TCP communication.
* Get-file: Reading file from the system and load it into the local machine. This function is also based on TCP communication since the TCP is much more stable than UDP. During reading progress, this function will compare several replicas information through analyzing the timestamp. The most recent one will deliver the file to the query your query machine.
* Delete-file: Sine every machine has other machines’ file information, the current machine will send delete message to others when receiving delete command from the users. In other words, the related file will be delete from the local machine when each machine receives the delete message. At the same time, the metadata in global\_file\_info and local\_file\_info should be removed from the dictionary.
* List file saving directory: For every put-file function, the replicas file information should be sent to all other machines, which means every machine define a dictionary to save replicas information.
* List file name in local machine: Whenever receiving a writing request and deciding to agree it, the machine will save the file name into a local dictionary.

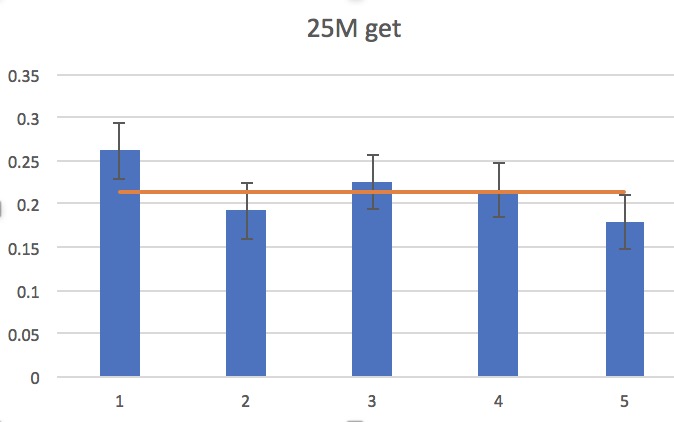
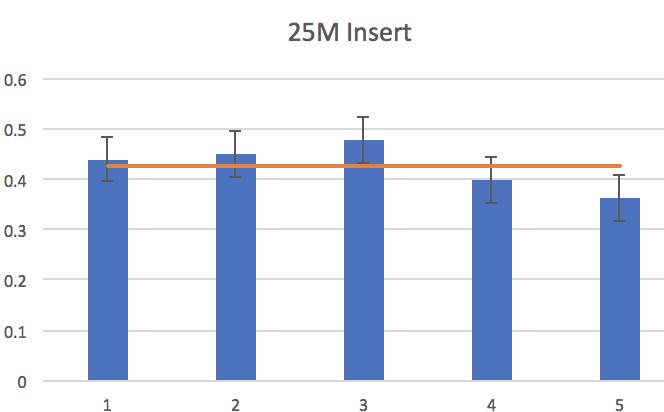
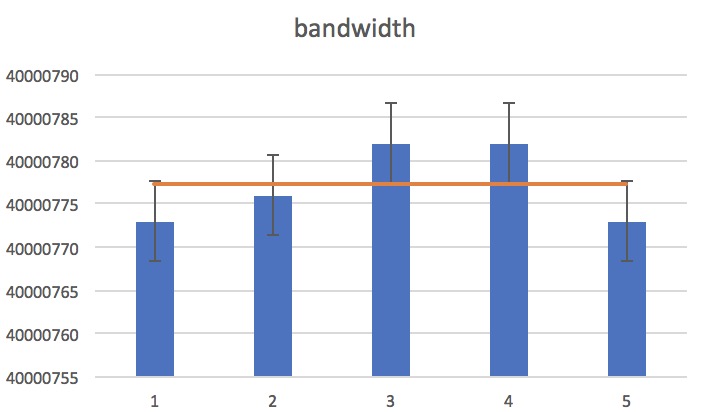
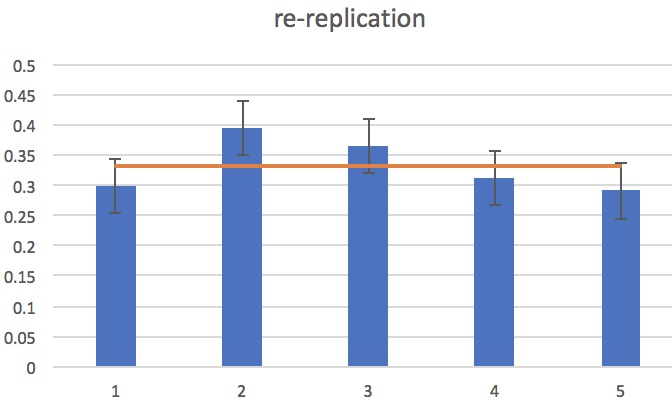
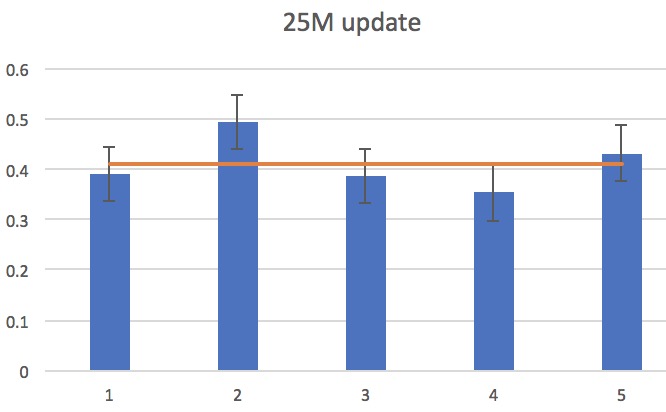
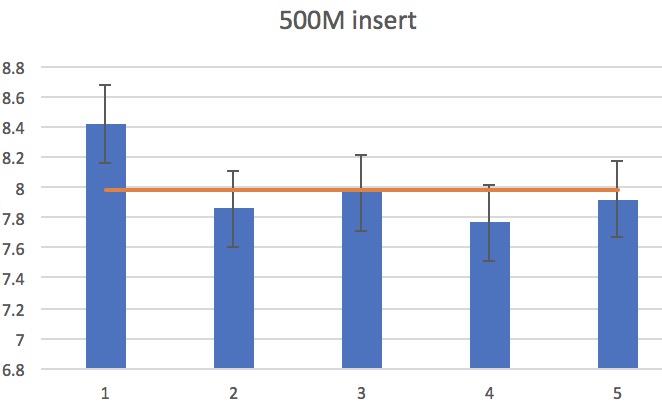
Except these upwards function, SDFS should deal with machines illegal exit and write-write problems. Whenever a machine which has certain replicas illegal quits from the system, the other machines which has the same replicas will elect a leader to send the replicas to a random machine which exist on the ring. For the write-write condition, whenever the write-file request happens on a same machine and the time gap smaller than 1 min, the current machine will ask the users to decide whether to rewrite the file.

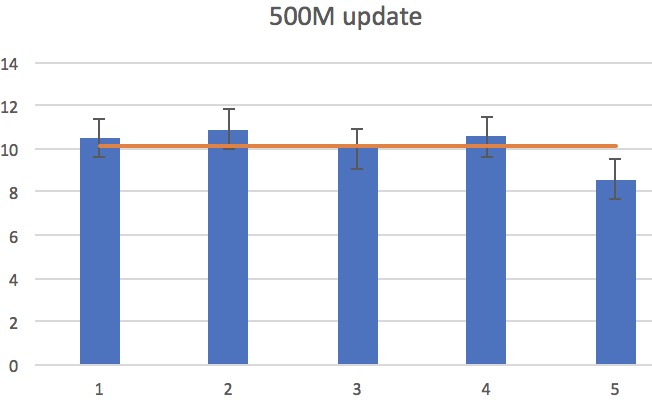
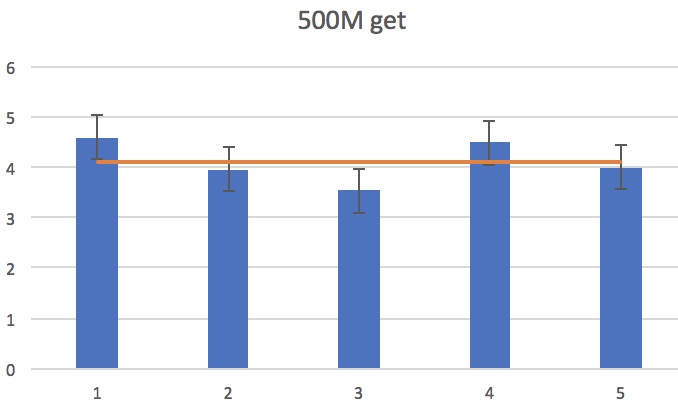
# Data:

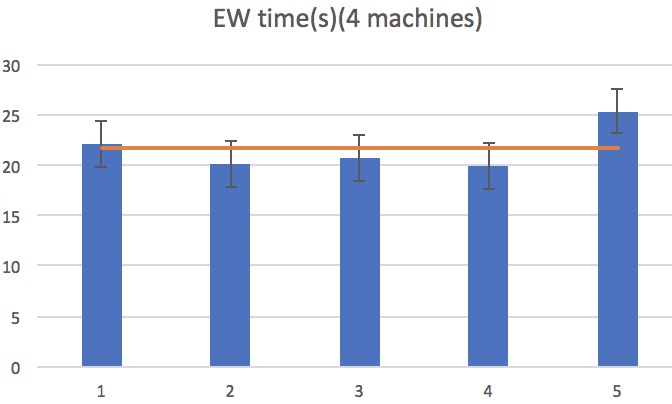
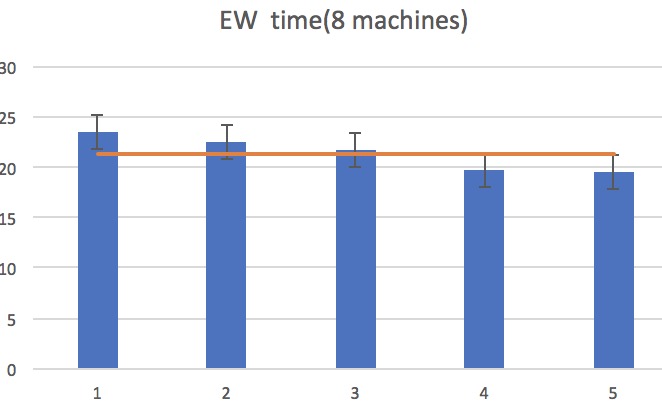
 





Above is what we expected: get/set time will increase if file size increases, but will not change significantly with more machines given large files. Write-write conflict detection should be fast, and message overhead should be small.