CS 5204 Project Final Report
A Feasible Study of MPI-IO on Top of HDFS

Luna Xu (xuluna@cs.vt.edu) Adam Binford (adamq@vt.edu)

November 30, 2014

1 Team member

Luna Xu (xuluna)
Adam Binford (adamq)

2 Introduction

This project aims at exploring the existing methods such as fuse-dfs [4] and nfs-dfs [1] to run MPI-IO applications on top of HDFS [7] directly. By measuring the feasibility and performance of read and write operations, we give insight for exploiting HDFS as underlying shared file system for MPI IO intensive applications.

In order to support such IO operations seamlessly and transparently without the overhead of multiple middle-layer overhead, we provide HDFS MPI-IO library using the native C lib [3] provided by HDFS project. Finally, we compare our implementation with the existing ones mentioned above.

3 Project Progress

We have finished investigating the possible ways to mount HDFS as a regular file system that can be interacted with by any file I/O. We got the throughput for manual copy, native NFS as the ideal performance, fuse-dfs throughput for parallel read. However we could not perform parallel write using fuse-dfs. Table 1 shows the errors we encountered during our tries. We tried using `MPI_File_write_at` where each process holds an individual file pointer, as well as `MPI_File_write_shared` where all processes hold a shared file pointer. We open the file using different mode and with the combinations we get mainly two errors. The error we get from the `APPEND` mode is reported in the MPI program side, others are shown in the fuse-dfs side. Another method that we explored is Native HDFS Fuse [6], which utilizes only protobuf to communicate with Namenode directly. Hence no fuse or native lib is involved. However, the program dumped a segmentation fault when we tried to run. HDFS-NFS solution is also not successful nor desirable because either it has requirements for specific (2.3.0) Hadoop version [1] or it only supports the cloudera distribution of Hadoop [2].

We are now focusing on creating a library to hook MPI [5] I/O function calls to use the HDFS native library to interact with HDFS. Our goal is to allow unmodified MPI applications to interact with HDFS by simply loading our library at runtime. So far we have successfully hooked MPI functions at runtime, and verified our functions were being called. Additionally, we have read from and written to files in our running HDFS using the HDFS native library. When reading a single file from multiple processes, we have observed an increase in bandwidth when increasing processes. This confirms that multiple processes can read from the same file at once using the native library. To complement this work, we have developed scripts to compile and run these HDFS native library applications easily.

4 Future work

The final steps we have to do are implementing the necessary MPI I/O functions in our hooking library to use the HDFS native library as the file I/O method. We have already done each of these pieces individually, hooking and using the native library, we simply must combine them. The

Function	Mode	Error
MPI_File_write_at	CREATE|RDWR	cannot open an hdfs file in O_RDWR mode
MPI_File_write_at	CREATE|WRONLY	cannot open an hdfs file in O_RDWR mode
MPI_File_write_at	WRONLY	cannot open an hdfs file in O_RDWR mode
MPI_File_write_shared	WRONLY	cannot open an hdfs file in O_RDWR mode
MPI_File_write_shared	APPEND	file open. code: 201388309

Table 1: Error codes for parallel writes on fuse-dfs.

hooking functions need to be able to use the parameters they are given to seamlessly work with HDFS without the MPI program knowing anything is different.

Additionally, we must find out if it is possible to implement some support for writing to HDFS through the hooked MPI routines. The native library only allows appending to a file, and only one thread can access a file for writing at one time. We must either modify the behavior of the I/O of the MPI program or set restrictions on what MPI programs running on HDFS are allowed to do. Finally, we must simplify the scripts required for our solution to work to put as small of a burden on the user as possible.

References

[1] Hdfs nfs gateway. http://hadoop.apache.org/docs/r2.3.0/hadoop-project-dist/hadoop-hdfs/HdfsNfsGateway.html.

[2] hdfs-nfs-proxy. https://github.com/cloudera/hdfs-nfs-proxy.

[3] Libhdfs. http://wiki.apache.org/hadoop/LibHDFS.

[4] Mountablehdfs. https://wiki.apache.org/hadoop/MountableHDFS.

[5] Mpich. www.mpich.org.

[6] Native hdfs fuse. https://github.com/remis-thoughts/native-hdfs-fuse.

[7] Hadoop Distributed File System (HDFS). http://hortonworks.com/hadoop/hdfs/, 2014.