Towards Exploring Data-Intensive Scientific Applications at Extreme Scales through Systems and Simulations
CS554 – Data Intensive Computing CWID – A20402686

Summary of the paper

This paper describes the FusionFS user level file system which works on file level. Nowadays, applications are becoming data centric. Workloads are data intensive rather than compute intensive. Compute system has significantly improved over the time, but there was no any growth in storage system. To overcome this problem, FusionFS helps us so that storage system improves its performance.

Let us discuss about FusionFS file system. In this file system, every compute node is involved in metadata and data movement. A global namespace of file system is maintained which can be accessed by client. Here, data and metadata are not together. If any data resides on a node, its metadata does not necessarily reside on the same node, it can be at different node. If we consider architecture of FusionFS file system, it is consisting of login nodes, compute nodes, storage nodes and I/O nodes. Login nodes and storage nodes are connected to I/O nodes through ethernet. Compute nodes are connected to I/O nodes through InfiniBand. InfiniBand is high throughput and low latency network. Every compute node contains local storage, distributed metadata and files. To run the applications in FusionFS file system, POSIX interface is implemented. Using POSIX interface, there is no need to modify the legacy applications.

To manage the metadata, Distributed Hash Table (DHT) is maintained so that every user can have a view of all files in FusionFS. This global namespace gives user a metadata information of every file, be it at local node or at remote node. The global namespace is just a logical view for client. It is not existed in data structure. In the data structure, there are two fields like *addr* for storing node on which file resides, *filelist* for storing all entries in the directory. FusionFS supports three protocols i.e. TCP, UDP and MPI for communication purpose. Persistence of metadata is important in case of storage failure. It is achieved by periodically flushing metadata from cache to local storage which is persistent. Here, flushing of metadata is asynchronous so that high performance can be achieved. Metadata is replicated for fault tolerance. There are many manipulation operations like file movement, file open [fopen()], file write, file read and file close. To simulate the performance of FusionFS, FusionSim simulator is implemented. This simulator is consisting of the ROSS simulation tool and the CODES simulation system. Using these two techniques, performance of FusionFS is simulated. Simulation system has four layers (application, file system, network and infrastructure).

If we evaluate the performance, FusionFS is better than GPFS in terms of metadata rate and I/O throughput. Real world applications where FusionFS helps to reduce I/O cost are PlasmaPhysics, Turbulence, AstroPhysics and Basic Local Alignment Search Tool (BLAST).

How is this work different than the related work?

Some systems are related to FusionFS file system such as provenance tracking, GPU accelerated erasure coding, cooperative coaching and file compression.

Some shared and parallel file systems such as Network File System (NFS), General
Purpose File System (GPFS), Parallel Virtual File System (PVFS) and Lustre consider that storage nodes are less than the compute nodes, and compute resources does not know the location of data in underlying storage system, which is an unbalanced architecture for data-intensive workloads, But FusionFS manages all these points efficiently, hence its architecture is balanced. Some file systems contain blob store which maintains metadata server which gives metadata to nodes. But FusionFS distribute all metadata without any chance of failure and has greater scalability. There is one more feature i.e. co-location of compute and storage resources. There is lot of work has been done on

data I/O with general rules. Systems like Nectar manages data and computations automatically using general rules of data locality. But FusionFS is optimized for workload which is write intensive.

Identify the top 3 technical things this paper does well.

- **Decoupling of metadata and data-** It allows different strategies to be applied on data management and metadata.
- **FusionSim Simulator –** This simulator performs well as compared to other systems.
- **I/O Throughput –** Read and Write throughput of FusionFS is much better than that of GPFS and Amazon EC2 Cloud.

Identify 3 things the paper could do better.

- FusionFS has been criticized for its performance overhead. In ext4 file system needs only two context switches between user and kernel. But FusionFS needs 4 switches, 2 between caller and VFS and another 2 between libfuse library. These 4 switches can cost more as compared with another file system.
- During the file read operation, remote reading should be completely avoided as it costs more.
- FusionFS should be improved to be used in applications like Turbulence and AstroPhysics.

If you were to be an author of a follow up paper to this paper, what extensions would you make to improve on this paper?

If I was an author of this paper, I would have included the method to avoid remote file read operation. I would have also added the process for managing the data and metadata more efficiently.