Reading Assignment 2

**SUMMARY**

In current HPC, there are two types of nodes viz Data node and Storage nodes. These nodes communicate by means of a shared network. Although the current supercomputers are extremely fast and are more penchant towards computation or computational intensive applications the need for data intensive applications is much needed as there was a boom in the field of big data. Due to these data intensive applications there are various bottlenecks and loos of efficiency caused above a certain level such as data beyond petabytes or data in exabytes. Despite intensive research on new HPC architecture on collocating node-local storage with compute resources, none of the real systems have any persistent storage on compute node. The creators of the fusionFS proposed a new architecture which is more concentrated towards data storage rather than compute intensive applications.

There are various kinds of other file system architectures i.e HDFS, GPFS,PVFS etc. which are used for big data applications or rather datacentric applications. But the storage systems optimized for data centers or cloud may not actually scale for high performance computers. First, the storage system on HPC nodes needs to support intensive metadata operations. Second, file writes should be optimized for the distributed storage on HPC nodes. The fault tolerance of most large-scale HPC systems is achieved through some form of checkpointing. This bottleneck was overcome by the FusionFS architecture which was actually designed for exabytes or exascales of data. To achieve maximum concurrency, FusionFS distributes its metadata to all the available nodes. Every client of FusionFS optimizes file writes with local writes which makes I/O throughput scalable. In FusionFS, remote storage mainly utilized for achieving the purpose.

The experimentation was done on HPC’s such as IBM blue gene where fusionFS was deployed. 16K nodes of an IBM Blue Gene/P supercomputer were heavily accessed by a variety of benchmarks and applications. FusionFS provides outstanding 2.5TB/s aggregate I/O throughput on 16K nodes which is better than other popular filesystems such as GFS, PVFS, and HDFS. This indicates FusionFS effectively works with today’s supercomputers. However, in this paper author(s) focuses on the scalability of FusionFS to Exascale. To test the scalability to exascale, authors develop FusionSim simulator. The result of simulations shows linear scalability of I/O performances with 329 TB/s throughputs on 2-million nodes and FusionFS viability for Exascale.

**How is this work different than the related work.**

FusionFS is said to be much faster as compared to other filesystems. Although for a filesystem to work on an HPC it must be POSIX compliant where POSIX is portable Operating System Interface. POSIX should be supported for the sake of backward compatibility.HDFS, doesn’t provide support to POSIX whereas in FusionFS, the POSIX interface is implemented with the FUSE framework. So, legacy applications can directly run on FusionFS without any modification but to run on HDFS some modifications are required.

Although HDFS is quite fast for current workloads and has the structure of a conventional centralized metadata. This design doesn't meet the requirement of many HPC applications which deals with large number of metadata operations. A single metadata server would easily become bottleneck in these metadata intensive applications and are not extremely scalable. Future storage systems for high-end computing should support distributed metadata management. FusionFs was designed to provide High scalability as well as concurrency( wrt to metadata).

**Top 3 technical things this paper does well**

1. **Fusion Simulator :** Built to test the workloads at exascale.
2. **File movement :** Use of FDT or fusion data transfer based on UDP based data transfer. Thus providing high reliability and congestion control.
3. **Metadata Management :** In FusionFS, Metadata and data are completely decoupled. This decoupling allows FusionFS to apply different tactics to manage metadata and data.

**Three things the paper could do better**

1)In FusionFS, POSIX is implemented with FUSE framework. To avoid the performance overhead from FUSE, FusionFS also provides a user library for applications to directly interact with their files. The downside of this approach is the lack of POSIX support, indicating that the application might not be portable to other file systems, and often needs some modifications and recompilation.

2)FusionFS works at the file level, thus chunking the file is not an option for file write operation. The chunk-level data movement of files is important while working on the large file.

3)Though FusionSim shows below 4% error rate for 16K nodes, running FusionFS on physical nodes can result differently from simulator results.

**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?**

1. Instead of spinning drives used at data centers SSD’s can be used as they do not have any moving parts and are extremely fast and reliable. They also support over provisioning and occupy physically less space.
2. Use of optic cables or Li-fi hardware to minimize the power cost.