Rapid Replication of Multi-Petabyte File Systems (Extended Abstract for SC15)

Justin Sybrandt
Grove City College
NERSC Student Researcher
Email: SybrandtJG1@gcc.edu

***Abstract*—As file systems grow larger, tools which were once industry standard become unusable at scale. Today, large data sets containing hundreds of millions of files often take longer to traverse than to copy. The time needed to replicate a file system has grown from hours to weeks, an unrealistic wait for a backup. Distsync is a new utility that expedites this process by quickly updating an out-of-date file system replica. By utilizing GPFS policy scan features, distsync finds changed files without navigating between directories. Distsync then parallelizes work across multiple nodes, maximizing the performance of a GPFS. Distsync has shown to be more efficient than existing methods and is expected to be used at NERSC to replicate file systems of over 100 million files and over 4 petabytes.**

I. Introduction

The National Energy Research Scientific Computing Center (NERSC) provides computational resources to the scientific community. Currently NERSC is preparing to move locations and needs to relocate its data safely and quickly. Moving locations, maximizing data availability, and improving disaster management are all cases where an organization should fully replicate huge file systems.

Recently, NASA developed tools which increased the rate of large file transfers. Shift [1] and mcp [2] can split the work of copying files over many cores (in the case of mcp) and many nodes (in the case of shift).

Unfortunately these tools do not perform well at NERSC's scale. Shift, in particular, spends a considerable amount of time traversing the file system, often spending more time finding files than moving them. To further complicate matters, shift does not preserve hard links or remove files at the destination that did not occur in the source.

Distsync is a new tool which attempts to provide a faster solution to file system syncronizaiton. It utilizes IBM's GPFS policy scans to quickly query the file system and produce large file records [3]. It determines what updates need to happen and splits that work between various job files. The distsync manager then takes the jobs and distributes them between its workers.

II. Architecture

First, the job generator compares GPFS policy scans to determine which files have changed since the last sync. Different lists of paths are created, called job files. The length of each file is limited to 10,000 paths in order to stay below the GPFS *MaxStatCache* value of 50,000. Limiting job file size ensures that the file stat cache is better utilized, greatly increasing performance. That being said, jobs relating to directories are not split; while time required to create or remove directories is nominal the effort needed to properly split and schedule directory jobs is very expensive.

Secondly, the Distsync manager sorts job files and spawns workers. This component schedules jobs, assuring that files, directories, and links are created in the right order. The manager tracks the status of each job, and makes sure that worker downtime is minimized.

Lastly, when a worker receives a job it communicates with the manager and runs system tools to complete its work. For example, a worker given a file copy job will run multiple parallel instances of mcp to transfer files from the fresh to the stale file system. The worker tracks each tool's instance and logs any errors it finds.

III. Conclusion

Due to hardware limitations during NERSC's move, the initial tests of distsync have been run on approximations of real file systems. The dummy systems have consisted of about one million files totaling around one hundred gigabytes. Although this is a small fraction of the actual scale of NERSC's GPFSs, initial results are very promising.

Still, there are cases where distsync performs poorly, most of which involve single long running jobs. For example, a large directory creation job can lead to workers idling if the only remaining work depends on the creation of directories. This case is rare unless distsync is being used to make the initial copy of a file system. Although distsync's performance is comparable to similar tools, it was not intended for that case.

Soon the technical limitations at NERSC will be resolved; file systems are currently being created to allow distsync to run on NERSC's largest, 4PB file system. Assuming the replication happens fast enough, distsync will be further developed and put into production at NERSC.

References

[1] P. Z. Kolano, "High performance reliable file transfers using automatic many-to-many parallelization," in *5th Wkshp. on Resiliency in High Performance Computing*.

[2] P. Z. Kolano and R. B. Ciotti, "High performance multi-node file copies and checksums for clustered file systems," in *Proc. if the 24th USENIX Large Installation System Administration Conf.*

[3] F. Schmuck and R. Haskin, "Gpfs: A shared-disk file system for large computing clusters," in *FAST '02 Proceedings of the 1st USENIX Conference on File and Storage Technologies* (editor, ed.), no. 19, USENIX Association Berkeley, CA, USA.