Comparing The Performance of Different Parallel Filesystem Placement Strategies

Esteban Molina-Estolano (student), Carlos Maltzahn, Scott Brandt, *University of California, Santa Cruz*; John Bent, *Los Alamos National Laboratory*

January 29, 2009

The different placement strategies used by parallel filesystems to select storage nodes for chunks of data can have considerable effects on performance. We are using trace-driven simulation to compare these placement strategies under different workloads.

We examine the placement strategies used by three different filesystems: Ceph, PanFS, and PVFS. Ceph and PVFS use calculated placement: with a small, constant amount of per-file metadata, they calculate which storage nodes to use for the different chunks of a file; PVFS uses a round-robin placement strategy, and Ceph uses the CRUSH pseudorandom placement function. [2] The RAID placement strategy used in PanFS also calculates the locations of individual chunks, but stores somewhat more placement metadata: it selects and records one or several RAID groups of storage nodes per file. The filesystems also differ in default chunk size and redundancy strategies. Ceph uses a 4 MB chunk size, while PanFS and PVFS use 64k. Ceph uses replication for redundancy; PanFS uses RAID 5; and PVFS uses no redundancy across storage nodes.

To compare the placement strategies, we use discrete-event simulation. We use traces from different workloads to drive simulated client nodes, which use the selected placement strategy to send I/O requests to the simulated storage nodes. Currently, we primarily use real and synthetic I/O traces from the checkpointing phase of scientific computing workloads. We also have webserver traces from an ISP, and we plan to acquire traces from data-mining workloads.

Our preliminary results examine workload balance across the cluster, under the three placement strategies, for three synthetic scientific computing checkpointing workloads from LANL. We also examine the balance effects of normalizing chunk size across the filesystems and turning off redundancy mechanisms. While workload balance alone is far from a good measure of overall performance, these results do show some tradeoffs between balance, data safety, and overall performance.

To measure balance, we divide the simulation run into 1-second intervals. For each interval, we divide the mean load per storage node by the maximum load, giving us a balance metric where 1.0 represents perfect workload balance across all nodes.

For these three workloads, our preliminary results show the PanFS and Ceph placement strategies to be comparably balanced; except with an interleaved write workload, where Ceph's is better balanced. The PVFS round-robin placement strategy is the best balanced, partially because of its small chunk size; reducing the Ceph chunk size from 4 MB to 64 KB, to match PVFS, improves balance considerably.

We also ran simulations with versions of the Ceph and PanFS placement strategies that had redundancy mechanisms turned off. This made little difference in terms of balance, with one exception: switching the PanFS placement strategy from RAID 5 to RAID 0 removed contention for parity blocks in the interleaved write workload, improving balance considerably.

Our simulator's performance model for storage nodes is currently rudimentary. To measure performance, instead of simply balance, the next stage in our work will improve the simulated storage nodes. We will add a buffer cache and prefetching, a local filesystem, and use DiskSim.

Since we are comparing the placement strategies of the three filesystems—not the filesystems in their entirety—we are not aiming for a complete simulation of each filesystem. Our ultimate goal, rather, is to understand the strengths and weaknesses of different placement strategies; and how other layers in the filesystem interact with the placement strategies. We are particularly interested in compensating for disadvantages of different placement strategy disadvantages (e.g. via caching or prefetching strategies).

References

[1] Sage A. Weil, Scott A. Brandt, Ethan L. Miller, Darrell D. E. Long, and Carlos Maltzahn. Ceph: a scalable, high-performance distributed file system. In *USENIX'06: Proceedings of the 7th conference on USENIX Symposium on Operating Systems Design and Implementation*, pages 22–22, Berkeley, CA, USA, 2006. USENIX Association.

[2] Sage A. Weil, Scott A. Brandt, Ethan L. Miller, and Carlos Maltzahn. Crush: controlled, scalable, decentralized placement of replicated data. In *SC '06: Proceedings of the 2006 ACM/IEEE conference on Supercomputing*, page 122, New York, NY, USA, 2006. ACM.