Fault Isolation: Dynamically Re-arranging Data to Increase Availability

Avani Wildani, Ethan L. Miller
University of California, Santa Cruz
{avani,elm}@cs.ucsc.edu

I. Introduction

As traditional media is progressively replaced by digital files stored in the cloud, our cultural mindset is shifting from considering digital data to be ephemeral to it being the primary, and often only, store of vital information. We are also seeing the first generations that record their entire lives and expect this data to be digitally stored perpetually, inexpensively, and searchably, creating massive data stores with both primary and archival usage patterns [2], [5]. We define the workload that this sort of data creates as "archival-by-accident."

In an archival-by-accident system, there are dynamic pockets of primary use in a storage system that otherwise exhibits traditional secondary or even tertiary workload characteristics. To handle these primary pockets on large systems over a long period of time, we must create a system with low cost, high reliability, and high availability. This availability requirement is the key addition our work imposes on traditional archival systems. The goal of this project is to exploit patterns in large data to co-locate working sets and thus improve system availability.

Improving availability is increasingly important as storage systems blossom in size and expected time of data retention. A 100PB system that has no failures 99.99999% reliability against data loss will still lose over 100 GB on average every hundred years and have to rebuild many terabytes more. Depending on the reliability scheme, certain failures can cause very long rebuilds that have a large impact on system availability. Rebuild time is also a function of the size of the components of the system. As these components increase in size, the availability loss caused by rebuilding data increases accordingly. Systems typically place data evenly across disks in order to distribute and thus dilute the impact of failure events. This data placement assumes that all data has identical probability of being accessed. While this assumption suffices for traditional archival systems, archival-by-accident systems have shifting subsets of data in active, primary use. For example, suppose a storage system contains the equal size inputs for ten experiments striped evenly across five disks. If one disk fails, all ten projects are stalled by the rebuild since they each have a small percentage of missing data. If, on the other hand, the data was arranged such that each experiment had the files it actively needed co-located, only two experiments would be stalled during rebuild while the other eight would only suffer reduced bandwidth.

II. Approach

To isolate faults to areas with lower probability of access, we need to quickly and reliably identify the pockets of data with primary access characteristics. In our previous work, we have found that it is possible to group data in large systems such that the groups have predictive power, meaning that group membership implies a conditionally dependent probability of access within a given period of time. We further believe that these groups can correspond to real-life working sets for applications, users, or projects. We define the "total system impact" of a failure event as the total delay encountered by all data groups as a result of the access requests that can not be completed because a disk is failed or rebuilding. By combining existing reliability measures with selectively laying out data according to group membership, we can isolate failures so that they impact fewer users or projects, resulting in a lower total system impact and an increase in the perceived availability of the system. We hypothesize that it is better for net system availability to have one project or working set severely impacted by occasional failure events than to have many projects slightly impacted. We rationalize this with a base case of a failure occurring and the affected data rebuilding before a single access to the data in question takes place. Here, there is zero productivity cost to the system; it becomes the proverbial silent tree falling in a forest. By placing data strategically, we hope to create more lightly accessed areas to isolate faults in.

III. Status

We are testing this theory with a failure injection simulation run over an actual trace of grouped data, with which we will compare the total system impact of disk rebuilds with and without grouping to the overall time the system is rebuilding. The purpose of doing the comparison this way is to track how much grouping increases the proportion of rebuilds with no system impact. We use rapid scrubbing to detect these failures for the test case, though in a real system there is no need for scrubbing to occur more frequently than is needed for reliability. Our simulator also takes into account both the increased risk of failure with every disk spin-up and the uniformly distributed constant risk of device failure to insert failures into the system.

After this research is completed, we expect to be able to show the effect of groupings on total system impact. We also intend to keep track of the impact per group and the effect of scrubbing frequency on these numbers. Reliability is a concern: by preferentially placing related data on the same physical medium, we increase the chance that a failure that causes data loss will significantly impact a given group. This is especially relevant as rebuild time is frequently traded for higher reliability. Grouping data offsets some of this rebuild cost while increasing per-project loss exposure. The overall reliability should remain unaffected, as very active data will still be served out of cache and different replicas are not placed identically. We are experimenting with different underlying reliability schemes to balance replication versus erasure coding. Our other main concern is that groups that are forced to rebuild will spend an unacceptable amount of time unavailable. This is fairly unlikely to be a problem as the rebuild time for rebuilding different sizes of data on a single disk has shown to be relatively constant [6]. significantly reduce the effective usability of most system since the difference in rebuild time for rebuilding one data block or rebuilding several on the same disk is fairly constant [6].

References

[1] T. Lohman. SKA telescope to provide a billion PCs' worth of processing. Computerworld, 2009. http://www.computerworld.com.au/article/319128/ska_telescope_provide_billion_pcs_worth_processing_updated_/.

[2] D. McCullagh. Why no one cares about privacy anymore. 2010. http://news.cnet.com/8301-13578_3-20000336-38.html.

[3] Thomas Schwarz, S.J. and Ethan L. Miller. Store, forget, and check: Using algebraic signatures to check remotely administered storage. In *Proceedings of the 26th International Conference on Distributed Computing Systems (ICDCS '06)*, Lisboa, Portugal, July 2006. IEEE.

[4] C. Staelin and H. Garcia-Molina. Clustering active disk data to improve disk performance. *Princeton, NJ, USA, Tech. Rep. CS–TR–298–90*, 1990.

[5] J. Urquhart. Does the fourth amendment cover 'the cloud'? 2008. http://news.cnet.com/8301-19413_3-10436425-240.html.

[6] A. Wildani, T.J.E. Schwarz, E.L. Miller, and D.D.E. Long. Protecting against rare event failures in archival systems. In *Modeling, Analysis & Simulation of Computer and Telecommunication Systems, 2009. MASCOTS'09. IEEE International Symposium on*, pages 1–11. IEEE, 2009.