Exploring Copysets under Repeated Failures

Jayden Navarro
jaynavar@stanford.edu
Stanford University

Isaiah Brandt-Sims
isaiahbs@stanford.edu
Stanford University

ABSTRACT
For our final project, we present a reproduction and extension of one of the results from the paper “Copysets: Reducing the Frequency of Data Loss in Cloud Storage” by Asaf Cidon et al [2]. First, we present a recreation and limited simulated reproduction of Figure 6, which shows the probability of data loss when 1% of the nodes in a cluster fail due to correlated failures (e.g. power loss) under different replication schemes. Additionally, we highlight a mistake we discovered in one of the equations used to generate Figure 6 in the original work. Next, we present an extension to the research, exploring the probability of data loss under repeated failure events. Our findings show that Copyset Replication parameters that yield high probability of data loss with single failures can perform better than those that yield low probability of data loss with single failures, when repeated failures are introduced at a short time interval.

1 INTRODUCTION
In large scale distributed systems, node failures are a fact of life. For storage clusters, node failures are particularly pernicious as they can result in permanent data loss. To combat this, data is distributed across multiple nodes for redundancy, however, as is explored in [2], the replication scheme chosen has a significant impact on the data loss durability of the system when multiple nodes fail simultaneously. These simultaneous failures can occur when a correlated failure event affects the entire cluster, such as a power outage.

The number of nodes used to replicate each unique chunk of data is termed the replication factor (R) of the system, and the set of nodes that store a given chunk of data is called the copyset for that data. In typical storage clusters, each chunk of data is replicated to a random set of nodes, which leads to a large number of total unique copysets in the system. If nodes fail simultaneously, data is lost if one or more combinations of the failed nodes yields a copyset that data was replicated to. Thus, as the total number of copysets in the system increases, the likelihood of data loss after correlated failures goes up proportionally.

Given the above, it would seem the best strategy would be to use the fewest number of copysets possible, however there is a trade-off. When the number of unique copysets is small, each node has fewer peer nodes that it shares replicated data with, leading to slower recovery time after a failure event since there are less nodes to pull from. The scatter width (S) of a cluster is defined as “the number of nodes that store copies for each node’s data” [2], thus there is a direct relation between the scatter width and the recovery time of a given node. The paper introduces Copyset Replication, a scheme that balances this trade-off, yielding a low number of total unique copysets given any required scatter width.

Copyset Replication generates available copysets by permuting the list of nodes and then combining adjacent nodes to form \( \lfloor N/R \rfloor \) copysets of size R. The number of permutations (P) used to generate available copysets is a function of the scatter width, namely \( P = S/(R - 1) \). When a chunk is replicated to the cluster, it can only be replicated to one of the generated copysets, reducing the total number of unique copysets in the system. The paper showed that Copyset Replication yielded significantly lower probabilities of data loss compared to random replication and a straw-man solution, with only a slightly larger recovery time for similar scatter widths. We identified a mistake in the random replication equation used to estimate the probability of data loss under this scheme, however this did not affect the findings of the paper.

Noting the scatter width trade-off between probability of data loss and recovery time, we extended the research to explore the probability of data loss under repeated failures. The intuition is that a slower recovery time could lead to fewer recovered nodes between repeated failure events, causing the set of failed nodes to increase thus leading to a higher probability of data loss. We simulated the Copyset Replication scheme, failure events, and recovery periods, tracking the state of the cluster at each stage.

Our results show that over a short failure time interval (20 minutes), after five or more repeated failures the probability of data loss is actually higher with S=10 (Facebook Copyset Replication) compared to S=200 (HDFS Copyset Replication). We also show that a slightly larger scatter width (S=20) can yield significantly lower probabilities of data loss under repeated failures, while only sacrificing a minimal increase in probability of data loss under a single failure. It remains to be investigated whether this failure pattern has been observed in production environments, and how closely our system parameters match those of deployed clusters. Our intuition is that repeated failures at a short time interval would be highly unlikely, as datacenter operators would leave the cluster shutdown after observing two or three repeated failures.
Figure 1: recreation of Figure 6 from the paper, plotting the probability of data loss when 1% of the nodes fail for different replication schemes as the number of nodes in the cluster increases. Top graph uses original equations obtained from the author, while bottom graph uses the corrected random replication equation.

2 FIGURE 6 REPRODUCTION

In order to replicate Figure 6 from the paper, we contacted the primary author, Asaf Cidon, who was able to provide us with the equations used to generate the plot. Given these equations, it was trivial to replicate the figure (see figure 1). However, later on in the project we discovered a mistake in the random replication equation, namely that the number of total chunks in the system was being calculated as $N \times C$ rather than $(N \times C)/R$, where $N$ is the number of nodes in the system, $C$ is the number of chunks per node, and $R$ is the replication factor. As each chunk must be replicated $R$ times, the total number of unique chunks in the system can only be as large as the chunk capacity of the system divided by the replication factor. We reached out to the primary author again, who confirmed that this was indeed an error in the original equation. Plot (a) shows the original equations, and plot (b) shows the results with the corrected random replication equation (the top two lines). While the new equation does lead to a slightly less steep curve, it yields no affect on the findings of the paper.

Note that we use the term recreation to refer to our plots that were generated using the equations provided by the author, and reproduction to refer to the plots generated by our simulations.

Next, we worked on reproducing the figure using simulations of each of the replication schemes. However, we soon realized that the computational requirements for simulating random replication on a per-chunk basis were far too great to perform a full reproduction of the figure, so we instead opted to perform a small scale reproduction on limited parameters, and to compare our results to a plot generated by the equations over the same parameters (see figure 2). We also ran a larger simulation of Copyset Replication, up to

Figure 2: Comparison of equations and simulations with number of nodes between 0 and 2,000 in step sizes of 100, using 1,000 trials per data-point.
Exploring Copysets under Repeated Failures

10,000 nodes, to verify the data loss lines matched the equations (see figure 3). Our simulated reproductions were able to validate the accuracy of the equations (after correcting the random replication equation) to a certain extent, barring the noisiness of the samples due to limited trials (figure 2 only used 1,000 trials per data-point, but even with such few samples it took 27 hours to collect the data).

A significant limitation of our random replication simulations was that they did not account for node capacities, due to the computational overhead of tracking and updating the state on a per-chunk basis. Because of this our simulations did not faithfully simulate the cluster, and instead used the same simplistic model that the equations assumed, which may be the primary reason they so closely match the equations. Another simplifying assumption that was made in the simulations was that all available copysets generated by the Facebook random replication scheme and Copyset Replication scheme were used to replicate chunks of data.

However, this assumption should hold as the number of used copysets is $P(1 - e^{-\frac{M}{P}})$ where $P$ is the number of generated copysets and $M$ is the total number of chunks in the system. This is approximately $P$ if the number of unique chunks in the system is at least one order of magnitude greater than the number of generated copysets, which is the case for both of these schemes.

3 REPEATED FAILURES

The primary trade-off explored in the paper is that with smaller scatter widths the recovery time increases as there are less nodes to pull from on recovery. The goal of our extension was to expose an additional trade-off in the choice of scatter width, showing that as failures occur repeatedly within a short time interval, a lower scatter width can also lead to a higher probability of data loss due to the increased recovery time.

Figure 4 shows the probability of data loss under repeated failures. In our simulation, we used a cluster size of 5,000 nodes with each node storing 1 TB of data and using a 10 Gb/s connection with 5% of the network bandwidth available for recovering failed peers (the paper showed recovery utilization rates between 3.9-4.2%, so we conservatively chose 5%). The simulation used Copyset Replication with scatter widths of 10 and 200, which are the scatter widths used by the Facebook and HDFS Copyset Replication schemes respectively, shown in Figure 1. Additionally, we included scatter widths of 20 and 100 as comparison points. On a 5,000 node cluster, this yields probability of data loss of 0.8% for the Facebook Copyset Replication scheme and 14.5% for the HDFS Copyset Replication scheme under a single failure event. These numbers closely match the values for $S=10$ and $S=200$ at timestamp 00:00 in the plots shown in figure 4.

To plot the probability of data loss under repeated failures, we ran the Copyset Replication algorithm given the specified scatter width and number of nodes, which generated the copysets for the given simulation run. To simulate a failure event we randomly sampled 1% of the nodes and checked if the copysets generated by the set of failed nodes overlapped the copysets generated by the Copyset Replication algorithm. If there was overlap, this meant there was data loss, and if the sets were disjoint this meant there was no data loss. To simulate recovery between the repeated failures, we determined the time to recover each failed node, taking into account the number of alive peers it shared data with, the capacity and bandwidth limitations of the alive peers, the number of failed nodes each alive peer was helping recover, and the input bandwidth of the failed node (see figure 5). If this recovery time was below the failure interval we marked the node as alive, otherwise the node remained failed. We made the simplifying assumption that node recovery was atomic,
recovery time = \frac{\text{node capacity}}{\text{min}(\text{node bandwidth}, \text{peer bandwidth})}

\text{peer bandwidth} = \sum_{p \in \text{alive peers}} \frac{\text{recovery utilization} \cdot \text{node bandwidth}}{|\text{failed peers of } p|}

Figure 5: Recovery time of a failed node.

meaning that if a node was only partially recovered before the next failure it would need to restart recovery during the next recovery period. After the recovery period, another failure event was induced where 1% of the remaining alive nodes were failed, and the process was repeated.

Figure 4 plots (a), (b), and (c) show the isolated probability of data loss at each time period, only considering whether data was lost during the current failure. Plots (d), (e), and (f) show the probability of cumulative data loss at each time period, taking into consideration whether data was lost at any point up to and including the current failure event. For repeated failures every 10 minutes, the probability of data loss rapidly increases for all scatter widths as the interval is too small to recover the failed nodes in all schemes. For repeated failures every 20 minutes, we see in (e) that after five failure events S=10 has a higher probability of having lost data than all other scatter widths due to the latter schemes being able to recover all failed nodes after each failure event compared to the former. At intervals of 36 minutes and greater all schemes are able to recover all of their nodes, so the failure events become independent as is shown in (c). Despite S=20 yielding significantly lower probabilities of data loss after repeated failures at an interval of 20 minutes, it only has marginally higher probabilities of data loss under single (or independent) failures, as evidenced by (c) and (f).

When we inspect the 14 minute interval in figure 6 we see that both S=10 and S=20 perform worse than S=100 and S=200, implying that in this interval range the lower scatter widths are both experiencing compounding failures while...
the larger scatter widths are able to recover all of their nodes. To investigate this further, we plotted the probability of data loss after ten failures over a range of intervals to determine at which intervals compounding failures cease for each of the scatter widths. Figure 7 shows the results, which expose that both S=100 and S=200 cease to experience compounding failures around the 12-14 minute interval range, S=20 around the 14-16 minute interval range, and S=10 around the 34-36 minute range. The results validate our hypothesis regarding the behavior observed in the 14 minute interval.

It is also interesting to note that both S=100 and S=200 experience the drop-off in the same interval, and S=20 is not far behind. This is simply due to the fact that once the number of peers for a failed node reaches a certain amount, the aggregate outbound bandwidth of the peers exceeds the inbound bandwidth of the recovering node, yielding no further benefit as more peers are added past this point.

Our results show that there is benefit to choosing a scatter width that is just large enough to generate enough peers to saturate the inbound bandwidth of the recovering nodes. Under the system parameters used in these experiments, S=20 is close to this optimal value, and only yields slightly higher probabilities of data loss under single failures when compared to S=10 (see 00:00 data-points in figure 4, and non-compounding failures in (c) and (f)).

From the equations in figure 5, it is clear that the ideal scatter width occurs when the peer bandwidth is equal to the node bandwidth, which happens when $S = \frac{\text{average # of failed peers}}{\text{recovery utilization}}$. Note that for the numerator we are only considering nodes that have at least one failed peer, so the average can never be less than one. For our system, assuming an average number of failed peers of one, we get an ideal scatter width of 20. Since figure 7 shows that S=20 is smaller than the ideal value, this implies that the average number of failed peers is greater than one in our system. As the average number of failed peers is itself dependent on the scatter width of the system, this equation in present form only provides an estimate for the ideal scatter width given an expected average number of failed peers, upon which an experimental search for the ideal value can be conducted. We leave the finding of a closed form solution to the average number of failed peers (for alive nodes that have at least one failed peer) as an area for future work.
4 METHODOLOGY

To plot the equations and simulations, we built a data collection framework in Python that allowed us to standardize data collection for both the computed probabilities of data loss and the simulated probabilities of data loss. This allowed us to easily generate the figures using both the equations and simulations over the same parameters, which was useful when doing the reproductions. To run our simulations we utilized the Stanford Rice compute cluster, which allowed us to perform long running simulations in the background.

Significant time was spent optimizing the replication scheme simulations in order to collect the data we presented. As discussed previously, we originally attempted to perform per-chunk replication, taking into account all aspects of the modeled environment such as node capacities, but soon abandoned this approach. Instead, to simulate random replication we still performed a per-chunk replication, but to increase performance we added short-circuiting that returned as soon as data loss was detected, and also used integer identifiers for each possible copyset so that we could generate a single random integer and then check if this was a failed copyset. For Facebook random replication and Copyset Replication we made the simplifying assumption that all generated copysets were used to store data, which allowed us to avoid per-chunk replication which enabled much shorter data collection times compared to the pure random replication schemes.

5 LIMITATIONS

The primary limitation of our simulation was not being able to efficiently perform per-chunk replication. This handicapped our reproduction in two significant ways, first by preventing a full-scale simulated reproduction of Figure 6, and second by not allowing us to fully model a real storage cluster, like taking into consideration node capacities.

A limitation of our repeated failures simulation was that we did not account for node start-up times after failure, which in an ideal world would just shift the interval forward by a fixed offset, but due to non-uniform start-up times it would realistically yield different recovery times for different failed nodes, both due to their individual start-up time and the start-up times of their alive peers that they will pull data from on recovery.

We also had hoped to discover whether the failure patterns we simulated had been observed in production environments, however we were not able to acquire statistics on this topic. We communicated with Backblaze, a backup storage company, however they did not track statistics related to full cluster failure events.

Due to time constraints we were unable to perform the repeated failures analysis on the two other replication schemes explored in the Copysets paper. However, as they both yield more copysets for comparable scatter widths compared to Copyset Replication, we anticipate that the results would show the same trends but with higher probabilities of data loss across the board.

6 RELATED WORK

Previous work on data replication in storage clusters has primarily focused on constraining replications to meet scaling or topology requirements, like replicating chunks across racks or other types of fault domains in the pursuit of high-availability live upgrades and maintenance, or for downsizing clusters during lower usage periods for power savings [3].

As is noted in the paper, Facebook had previously worked on a custom replication scheme for their HDFS cluster which constrained replication to avoid data loss under concurrent failures [1], however this scheme yielded much higher probabilities of data loss compared to Copyset Replication for similar scatter widths.

7 FUTURE WORK

As discussed previously, due to the challenges we faced with per-chunk replication we were not able to fully model the constraints of a real storage cluster. An interesting area for future work would be to run these experiments on real storage clusters to determine whether the behavior observed in our simplified model could be verified on a real system.

Another interesting area for future work would be to explore the probability of data loss under correlated failures in cold storage environments, like those used by Backblaze and other data backup companies. In contrast to the full-copy replication used by the hot storage systems explored in this paper, cold storage clusters typically use erasure coding and parity drives to yield higher storage capacity. However, in these architectures they primarily use replication at a drive level rather than chunk level, greatly simplifying the complexity of the system and negating the large explosion of copysets explored under the context of this paper.

As was mentioned in section 6, previous work has been done on replicating data using specific schemes to meet other system needs. It would be interesting to explore the effect these schemes have on the probability of data loss under correlated failures, and whether new schemes can be generated that yield the benefits of Copyset Replication while meeting the existing replication requirements.

8 CONCLUSION

In this report, we presented a reproduction of Figure 6 from the Copysets paper, identified an error in one of the equations, and showed a simulated reproduction which reinforced the findings of the paper. Additionally, we identified that there
exists an optimal scatter width which minimizes the probability of data loss under repeated failures, and showed that even much larger scatter widths can yield lower probabilities of data loss than smaller scatter widths under specific conditions.

Our code is located at https://github.com/jaynavar/Copysets which includes a README with installation instructions and information on how to use the tools, as well as the data and figures used in our report.

9 ACKNOWLEDGEMENTS

We would like to thank the CS 244 teaching staff, Keith Winstein, Sachin Katti, and Bruce Spang, for their invaluable feedback and direction during this project, as well as for leading a fantastic course all quarter! In particular we would like to thank Bruce who helped us with many of the computational and combinatorial challenges we faced during the project. We are grateful to Ahin Thomas from Backblaze for coordinating with their team and pushing to get our questions answered. Finally, we would like to thank the primary author of the Copysets paper, Asaf Cidon, for graciously lending his time to help us understand the replication schemes, and for answering clarifying questions as the project progressed.

REFERENCES