

Combinatorial analysis of burst failures for large-scale cluster

Meng Wang

November 20, 2022

1 Setup

We consider a storage system of N drives such that $N = X \cdot Y \cdot Z$, where there are X racks in the system, each rack contains Y enclosures, and each enclosure contains Z drives.

Let $M = Y \cdot Z$, so M denotes the number of drives per rack.

In particular, we are considering ORNL Alpine system, which is composed of 39 racks. 38 racks have 8 enclosures each, and 1 rack has 4 enclosures. Each enclosure has 106 drives.

For simplicity, we assume the system contains 40 racks. Each rack contains 8 enclosures. Each enclosure contains 100 drives.

2 Total instances with fixed number of affected racks

Consider f failures happen in r racks.

We first choose r racks from all the X racks, which has C_X^r combinations.

Given r racks $1, 2, \dots, r$, denote $T(f, r)$ as the total number of instances for f failures to happen in r racks, such that each rack has at least one failure.

Consider the r -th rack. If it has i failures, then the rest $r - 1$ racks must have $(f - i)$ failures in total with each rack having at least one failure.

Therefore, we have the following recurrence relation:

$$T(f, r) = \sum_{\substack{1 \leq i \leq f \\ i \leq M}} C_M^i \cdot T(f - i, r - 1) \quad (1)$$

The base case is

$$T(f, 1) = \begin{cases} C_M^f & \text{if } 1 \leq f \leq M \\ 0 & \text{otherwise} \end{cases} \quad (2)$$

Therefore, we can compute $T(f, r)$ using dynamic programming, with time complexity $O(f \cdot r)$, and memory complexity $O(f \cdot r)$. An implementation can be found at: https://github.com/ucare-uchicago/mlec-sim/blob/main/src/theory/burst_theory.py#L30

The total number of instances is:

$$C_X^r \cdot T(f, r) \quad (3)$$

3 Survival instances under local clustered (RAID)

Consider $k_l + p_l$ local-only RAID SLEC. For easier deployment we assume $n_l = k_l + p_l$ is divisible by Z .

n_l drives in the same enclosure compose a RAID disk group. Therefore a rack contains $g = M/n_l$ RAID groups.

Denote $\eta(f, g)$ as the number of instances for one single rack to survive f failures in a rack containing g $k_l + p_l$ RAID groups.

For $\eta(f, g)$, we have the following recurrence relation (which is derived by considering what will happen if g -th group contains i failures):

$$\eta(f, g) = \sum_{\substack{0 \leq i \leq p_l \\ a \leq f}} \eta(f - i, g - 1) \cdot C_{n_l}^i \quad (4)$$

The base case is

$$\eta(f, 1) = \begin{cases} C_{n_l}^f & \text{if } 0 \leq f \leq n_l \\ 0 & \text{otherwise} \end{cases} \quad (5)$$

We can then compute $\eta(f_i, g_l)$ based on recurrence relation 4 and dynamic programming, with $O(f \cdot g)$ time complexity and $O(f \cdot g)$ memory complexity.

Given r racks $1, 2, \dots, r$, denote $S(f, r)$ as the total number of instances for $k_l + p_l$ local-only RAID to survive f failures to in r racks, such that each rack has at least one failure.

For $S(f, r)$, we have the following recurrence relation (which is derived by considering what will happen if r -th rack contains i failures):

$$S(f, r) = \sum_{\substack{1 \leq i \leq f \\ i \leq M}} \eta(i, g) \cdot S(f - i, r - 1) \quad (6)$$

The base case is

$$S(f, 1) = \begin{cases} \eta(f, g) & \text{if } 1 \leq f \leq M \\ 0 & \text{otherwise} \end{cases} \quad (7)$$

Since we have already computed $\eta(f, g)$, we can further compute $S(f, r)$ based on recurrence relation 6 and dynamic programming. The total time complexity $O(f \cdot (g + r))$, and total memory complexity and $O(f \cdot (g + r))$.

Here is an example implementation: https://github.com/ucare-uchicago/mlec-sim/blob/main/src/theory/burst_theory.py#L97

Therefore, the total number of survival instances in the whole system is:

$$C_X^r \cdot S(f, r) \quad (8)$$

Therefore, the probability of data loss under f failures on r racks for RAID is:

$$\text{RAID data loss} = \frac{C_X^r \cdot S(f, r)}{C_X^r \cdot T(f, r)} = \frac{S(f, r)}{T(f, r)} \quad (9)$$

4 Survival instances under local declustered parity

It's similar to local clustered erasure in Section 3, but now the size of the disk group is usually larger than n_l .

Suppose the size of the disk group is D , usually $n_l \leq D \leq Z$, where Z is the size of the enclosure.

If any disk group has more than p_l disk failures, then there is data loss.

So this time a rack contains $g = M/D$ disk groups.

We can then compute the data loss in the same way that we did for RAID. Therefore, the probability of data loss under f failures on r racks for local-only Declustered erasure is:

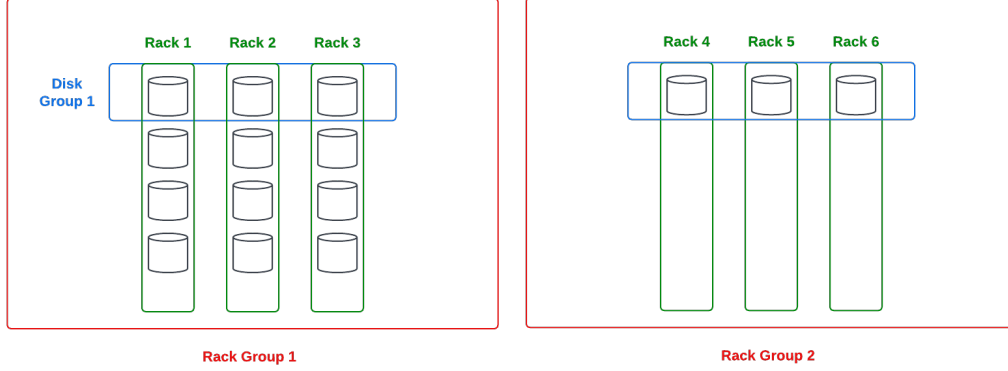
$$\text{Declustered data loss} = \frac{C_X^r \cdot S(f, r)}{C_X^r \cdot T(f, r)} = \frac{S(f, r)}{T(f, r)} \quad (10)$$

5 Survival instances under network clustered erasure

Consider $n_n = k_n + p_n$ network-only clustered erasure.

We first divide all the racks into $\alpha = X/n_n$ rack groups, each rack group with $n_n = k_n + p_n$ racks.

In each rack group, we further divide the disks into disk groups. Each disk group contains $n_n = k_n + p_n$ disks spread on n_n racks, with each disk on a different rack.



Denote $\eta(f, r, m)$ as the number of instances for one rack group to survive f failures in r racks in this rack group, with each rack contains m disks.

Thus there are m disk groups in the rack group: $1, 2, \dots, m$. Suppose the m -th disk group has i failures spread in i racks. Then disk group $1, 2, \dots, m-1$ should have $f-i$ failures in total. Suppose these $f-i$ failures spread in j racks. Then the i racks affected by disk group m and the j racks affected by disk groups $1 \dots m-1$ must overlap on $i+j-r$ racks. And there are $r-j$ racks which are affected by disk group m only.

If we consider of the possible values of i and j , we can derive the following recurrence relation for $\eta(f, r, m)$:

$$\eta(f, r, m) = \sum_{\substack{0 \leq i \leq p_n \\ i \leq r}} \sum_{\substack{0 \leq j \leq r \\ r-i \leq j}} C_j^{i+j-r} \cdot C_{n_n-j}^{r-j} \cdot \eta(f-i, j, m-1) \quad (11)$$

The base case is

$$\eta(f, 1, m) = \begin{cases} C_m^f \cdot C_{n_n}^1 & \text{if } 0 \leq f \leq m \\ 0 & \text{otherwise} \end{cases} \quad (12)$$

We can then compute $\eta(f, r, M)$ based on recurrence relation 11 and dynamic programming, with $O(f \cdot r \cdot M)$ time complexity and $O(f \cdot r \cdot M)$ memory complexity. Here is an example implementation: <https://github.com/ucare-uchicago/mlc-sim/blob/main/src/theory/netraid.py#L32>

Given α rack groups $1, 2, \dots, \alpha$, denote $S(f, r, \alpha)$ as the total number of instances for $n_n = k_n + p_n$ network-only RAID to survive f failures in r racks, such that each rack has at least one failure.

For $S(f, r, \alpha)$, we have the following recurrence relation (which is derived by considering what will happen if α -th rack group contains i failures affecting j racks):

$$S(f, r, \alpha) = \sum_{\substack{0 \leq i \leq f \\ i \leq M \cdot p_n}} \sum_{\substack{0 \leq j \leq r \\ j \leq i \\ j \leq n_n}} \eta(i, j, M) \cdot S(f-i, r-j, \alpha-1) \quad (13)$$

The base case is

$$S(f, r, 1) = \eta(f, r, M) \tag{14}$$

Since we have already computed $\eta(f, r, m)$, we can further compute $S(f, r, \alpha)$ based on recurrence relation 13 and dynamic programming. The total time complexity $O(f \cdot r \cdot \alpha)$, and total memory complexity and $O(f \cdot r \cdot \alpha)$.

An example implementation can be found at: <https://github.com/ucare-uchicago/mlec-sim/blob/main/src/theory/netraid.py#L76>

6 Survival instances under network declustered erasure

This is easy.

Consider $n_n = k_n + p_n$ network-only declustered erasure.

There is data loss whenever there are more than p_n affected racks.

Therefore:

$$\text{Net-declus Data loss} = \begin{cases} 1 & \text{if } r > p_n \\ 0 & \text{otherwise} \end{cases} \tag{15}$$

where r is the number of affected racks.

7 Survival instances under MLEC clustered

TBD

8 Survival instances under MLEC Declustered

TBD