Reasoning About SSD Write-Amplification - CSC443H1 Database System Technology – Niv Dayan

Pages and Erase Units. An SSD based on NAND flash memory consists of multiple pages, each approximately 4KB. Reads and writes take place at the granularity of pages. Pages are organized into erase units, and they must be written sequentially within an erase unit (consisting of hundreds to thousands of pages). To update a physical page, we must erase the entire erase unit that the page belongs to. Erase units have a finite lifetime: they can only be erased a certain number of times before they become too error-prone to store data reliably.

Out-of-Place Updates. SSDs are therefore designed to prevent having to erase and rewrite an entire erase unit every time we want to update the contents of an individual page. It does this by updating pages out-of-place. Specifically, it uses some metadata to mark the original version of the page as invalid, and it writes the new version of the page on some erase unit with free space. The SSD maintains a mapping table from the logical address of each phase to its physical address within the SSD. Overall, each page can have one of three states: free, valid, or invalid.

Garbage Collection. As updates take place, more physical pages within the SSD get marked as invalid. Eventually, as the SSD runs out of free space, we must reclaim space taken up by invalid pages to accommodate more updates from the application. The SSD does this by performing garbage collection. The garbage collector picks the erase unit with the highest number of invalid pages. It migrates any remaining valid pages into an erase unit with free space, and it then erases the target erase unit.

Write-Amplification. These garbage-collection operations induce write-amplification, the phenomenon whereby more physical work gets done within the SSD for any unit of logical work that the application is trying to do. Write-amplification disrupts performance and further reduces the lifetime of an SSD. It is therefore important to keep it low

Over-Provisioning. To limit write-amplification, any SSD is assigned more physical space than the logical capacity it exposes to the user. This extra capacity allows more invalid pages to accumulate before free space runs out. This means that in each erase unit, there is a higher proportion on average of invalid pages. For each garbage-collection operation, we, therefore, need to migrate less valid pages. This reduces write-amplification.

Quantifying Write-Amplification. Let x be the average fraction of valid pages in the target erase unit that we pick for garbage collection. Let us also assume each erase unit has y pages. This means that for every (1-x)*y invalid pages reclaimed during garbage collection, we must migrate x*y valid pages. Hence, the average number of pages we must migrate to reclaim one page is x*y / (1-x)*y = x / (1-x). Hence, for every page the application writes, we must physically write that page and in addition perform x / (1-x) writes in the background. This leads to a general write-amplification expression:

write-amplification per page written $= \left(1 + \frac{x}{1-x}\right)$

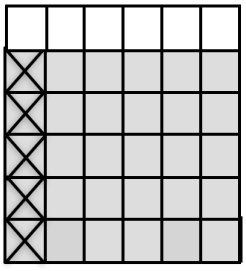
Write-Amplification from a User's Perspective. We usually like to express write-amplification as the total amount of physical space within the SSD that gets rewritten divided by the size of the update that the user wanted to make. If users make updates that are smaller than a page, an entire page still must be written as this is the minimum SSD write granularity. This further increases write-amplification.

For example, suppose the user updates 1KB chunks. For each 1KB chunk, a whole 4KB page has to be rewritten. For this to occur, x / (1-x) page migrations must have also happened in the background to have one free page available that we can write to. Hence, write-amplification is 4 * (1 + x / (1-x)) in this case, since for each 1 KB update, 1 + x / (1-x) 4KB pages must be rewritten in storage.

Generally, suppose that B data items fit into each page and that the user updates one random entry in each request. A general expression for write-amp from the user's perspective is:

write-amplification per entry written $= B \cdot \left(1 + \frac{x}{1-x}\right)$

Worst-Case Write-Amplification. The worst-case write-amplification occurs when every full erase unit in the SSD has the same number of invalid pages. In this case, the SSD is unable to pick an erase unit with especially few live pages left to migrate, and so garbage-collection overheads are the most expensive they can be. This worst-case is admittedly contrived, as usually there will be an uneven distribution of invalid pages across erase units (on this this later). However, this worst-case helps us to upper bound write-amplification. This can be visualized as follow. Each row corresponds to an erase unit with 4 pages. Gray means a valid page, white means a free page, and an X means an invalid page. In this example, each garbage-collection operation would have to migrate 5 pages to free one page.

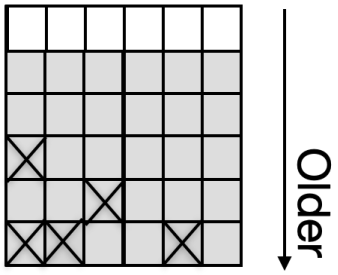


Let us denote the logical capacity of the SSD as L and the physical capacity of the SSD as P. In each full erase unit, a fraction of L/P of the pages is valid while 1-L/P is invalid. Hence, we can plug int L/P into our equation above to reason about worst-case write-amplification.

Worst-case write-amplification $= B \cdot \left(1 + \frac{\frac{L}{P}}{1-\frac{L}{P}}\right) = B \cdot \left(1 + \frac{L}{P-L}\right)$

For example, if we have an SSD with a logical address space of 800GB and a physical capacity of 1TB, and the user is issuing 4KB updates, then L/P = 0.8 and B=1. In this case, the worst-case write-amp is 1+0.8 / (1-0.8) = 5.

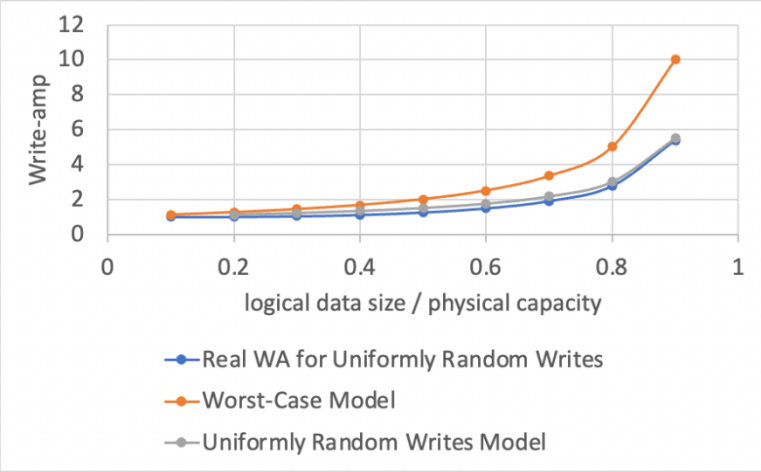
Write-Amplification Under Uniformly Random Writes. Let us now consider the case where the workload consists of uniformly randomly distributed writes. In this case, each page in the system is equally likely to be updated next, regardless of when it was written. In this case, erase units that were written a longer time ago will tend to have more invalid pages. Hence, the garbage-collector will generally be able to find erase units with fewer than L/P valid pages left.



So what is x in this case? In other words, what is the average number of live pages we must migrate in the erase unit with the least number of live pages left? In Section 4 of the following document, I show how to derive this precisely: https://arxiv.org/pdf/1504.00229.pdf. Section 4 in the following paper also provides a good approximation: http://www.vldb.org/pvldb/vol6/p733-stoica.pdf.

While the derivations in the above documents are precise, they use more complicated math that I would prefer to leave out of this course. I provide you with the following approximation instead. It is not formally derived, but as you can see in the following curve, it holds well in practice, as shown in the figure below.

Uniformly randomly distributed write-amplification $= B \cdot \left(1 + \frac{\frac{L}{P}}{2 \cdot \left(1-\frac{L}{P}\right)}\right) = B \cdot \left(1 + \frac{L}{2 \cdot (P-L)}\right)$



The Impact of Workload on Write-Amplification. In some workloads, the application sequentially writes a lot of data at once, and this data is then updated all at once. The SSD is likely to map pages from such large writes into the same erase units. When this data is then updated, many pages or even all pages within the same erase units become invalid at the same time. This makes it possible to perform reclaim erase units even more cheaply by migrating fewer or even no pages. If all pages in an erase unit are invalid, we can simply erase the unit with no page migrations and thus no write-amplification.

A pitfall, however, can occur if multiple applications are issuing such workloads at the same time. In this case, data from across different applications can become interspersed within the same erase units. This means that when some of this data is updated, not all pages within these erase units would become invalid at the same time. Hence, garbage collection may still take place to reclaim space, leading to write-amplification. This is precisely the problem we saw in the Research lecture that the Spooky paper strives to address.