Analysing an Effect of Buffer Capacity and Periodic Dirty Buffer Flush on Data Lifetime

Ilhoon Shin[1], Kyungah Lee[2]

[1] Professor, Department of Electronic & IT Media Engineering, Seoul National University of Science and Technology, Seoul, South Korea

[2] Student, Department of Electronic & IT Media Engineering, Seoul National University of Science and Technology, Seoul, South Korea

Abstract *— This study analyses an effect of buffer capacity and periodic dirty buffer flush on data lifetime on SSDs equipped with large internal buffer and obtains the following results. First, if the internal buffer exists, the proportion of short-lived data decreases. However, when the dirty buffer flush period is shorter than 30 minutes, even if the buffer size increases to 512 MB, most data still has a short lifetime. Second, when the dirty buffer flush period becomes longer (for example, one day), the proportion of short-lived data decreases rapidly as the buffer size increases. As a result, the accuracy of the existing lifetime prediction policy drops drastically. These results indicate that the lifetime prediction policy should be designed considering the presence of the internal buffer in SSDs and the period of dirty buffer flush.*

Keywords — *Buffer capacity, dirty buffer flush, data lifetime, SSD*

I. INTRODUCTION

NAND-based SSDs are replacing hard disks not only in PC and mobile computing environments, but also in the enterprise storage market such as servers. In servers, most data have very short lifetime [1-3]. That is, data are updated frequently in a short time. This means that SSDs do not need to retain data for a long time. Conventional SSDs write data in a deep write mode that guarantees a long retention time but is slow. But, if most data are short-lived, it is more beneficial in terms of the performance to write data in a shallow write mode that is fast but guarantees a short retention time [1, 2].

The important thing is to predict the data lifetime accurately. For example, if long-lived data is mistaken for short-lived data and written in the shallow write mode, it should be rewritten in the deep write mode before its retention time expires. This rewrite overhead incurs an additional NAND write and more frequent garbage collection, which results in hurting the performance and the lifetime of SSDs. Therefore, it is important to accurately predict the lifespan of the data. Especially, it is necessary to minimize to incorrectly identify long-lived data as short-lived data.

Meanwhile, there are three existing data life prediction policies. [1] judges all write requests from host as short-lived data. The overall accuracy of this policy is high because most data in servers are short-lived. The second method is to determine the lifetime based on the write request size. For example, in [2], if the size of a write request is 8 KB or less, it is classified as short-lived data. Lastly, [3] uses both the request size and the previous lifetime of the data. When data is first written, data having a size of 32 KB or less is classified as short-lived data. When data is updated, the future lifetime is predicted to be short if its previous lifetime is short.

Existing life prediction policies have a common limitation that they predict the lifetime without considering the existence of the SSD internal buffer. Typically, SSDs employ DRAM inside to maintain the mapping table for FTL [4], and a portion of DRAM is used as a buffer for NAND. With the buffers, write requests are primarily handled by the buffer, reducing the number of write requests issued to NAND. Therefore, it is expected that the data lifetime will increase compared to the environments without the buffer.

In fact, the existing study has shown that writing all host write requests in the shallow write mode in the SSD with the internal buffer significantly hurts the overall performance due to the rewrite overhead [5]. However, this study has a limitation that flushing the dirty buffers is not considered. That is, all write operations to NAND are limited to the case where the dirty buffer is evicted. However, in practice, the dirty buffers are flushed frequently due to the data synchronization requests by applications or kernel [6]. The dirty buffers are likely to be flushed by the synchronization request long before the dirty buffer is evicted. Therefore, it is necessary to reflect this dirty buffer flush and analyse the potential benefits of shallow write for the short-lived data.

This paper aims to analyze the lifetime distribution from the perspective of NAND when a SSD with a large buffer is used as a storage device for a server. Especially, the periodic dirty buffer flush effect is considered together. The remainder of this paper is organized as follows. Section 2 describes SSD internals and NAND flash memory characteristics. Section 3 describes the experimental environment and analyses the results of data lifetime distribution. Section 4 draws conclusions and future research issues.

II. SSD AND NAND FLASH MEMORY

SSDs connect multiple NAND chips in parallel to achieve large capacity and high performance. NAND is composed of pages and blocks, where a page is a basic unit of a read/write and a block is a basic unit of an erase operation. NAND is a kind of EEPROM and does not support overwrite operations. That is, data should always be written to a clean page. When an overwrite request occurs, data is written to a new clean page. Thus, the location of valid data is changed upon every write request, and FTL maintains a mapping table between logical sector numbers and their current data locations [8]. Since the mapping table is referenced upon every read/write request, it is maintained in the RAM inside the SSD. Of the RAM space, available space other than the mapping table is used as a buffer for NAND. Therefore, a large number of read/write requests are served through the buffer without accessing NAND.

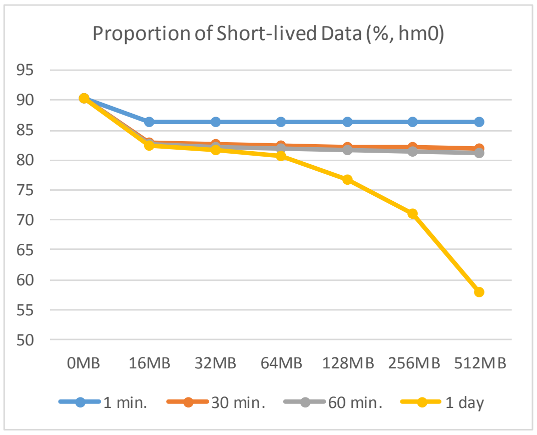
NAND flash cell expresses a value through the amount of electrons charged in its floating gate. For example, a SLC (Single Level Cell) expresses a state in which a floating gate is fully charged as 1 and an empty state as 0. In MLC (Multiple Level Cell), one cell expresses four values of 00, 01, 10, and 11. That is, the charge level is further subdivided to distinguish values. The problem is that the error rate increases because the charge level in one cell should be controlled more precisely. Moreover, as the cell size gradually decreases with the progress of semiconductor process technology, the margin of identifying each charge level becomes smaller.

Therefore, NAND write operation is performed in an ISPP (Incremental Step Pulse Programming) method [7] that repeatedly charges electrons by increasing the voltage step by step until the target electrons are charged for precise control, rather than charging electrons at a high voltage at a time. For precise control, the deep write method reduces the threshold voltage at each repetition, and as a result, it is slow but the number of the charged electrons can be accurately controlled. So, even if the electrons are naturally lost, the probability of the value being changed is small, and the retention time of the data is long. Conversely, if the threshold voltage is high, NAND write becomes faster, but it is difficult to precisely control the charge. Thus, the probability of changing the value increases as the electrons are naturally lost, which shortens the retention time of the data. Currently, NAND Manufacturers handle write operations in the deep write method for the long retention time. However, if the data lifetime is accurately predicted, we can shorten the latency of NAND write operations by applying the shallow write mode for the short-lived data [1, 2].

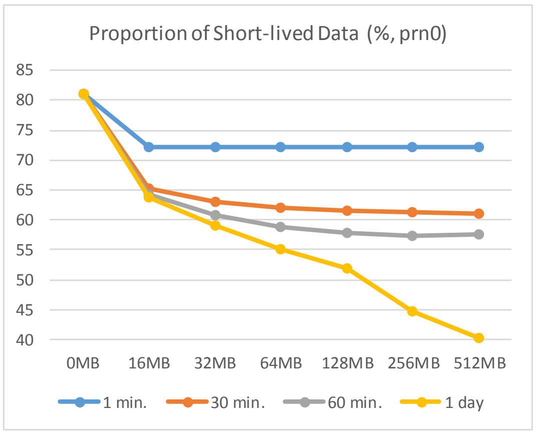
III. EXPERIMENTAL RESULT

Microsoft server traces [8] are used to analyse the lifetime distribution in SSDs with the internal buffer. The NAND page size is assumed to be 4 KB, and the internal buffer capacity is changed from 0 MB to 512 MB. The dirty buffer flush cycle is changed to 1 minute, 30 minutes, 60 minutes, and 1 day. Data having a lifetime of one day or less is evaluated as data having a short lifetime.

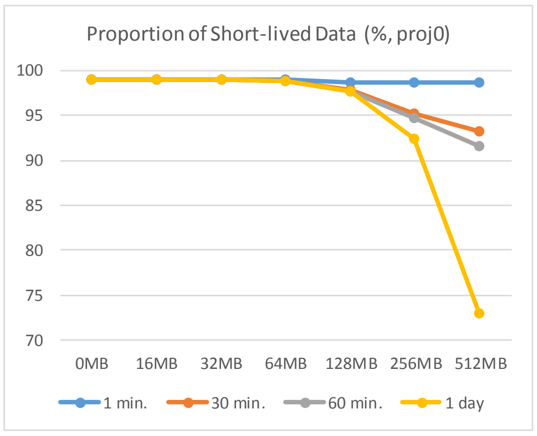
Fig. 1 (a) shows the proportion of short-lived data in hm0 trace. First, when the internal buffer is not used (0MB), short-lived data accounts for 90% of the total. That is, most data have a lifetime of less than one day. When the buffer is used, some write requests are served by the buffer, which reduces the proportion of short-lived data somewhat. When the flush cycle is 1 minute, short-lived data occupies about 86% regardless of the buffer size. Even if the buffer size increases to 512 MB, the proportion of short-lived data does not increase. Because the flush cycle is very short, the NAND write operations frequently occur due to the dirty buffer flush, and the host's write requests mostly generate NAND write operations by the dirty buffer flush. As the flush cycle increases to 30 minutes and 60 minutes, the proportion of short-lived data decreases to about 81-83%. However, even at this time, the buffer size does not significantly affect. The difference in the short-lived data ratio between the buffer size of 512MB and the buffer size of 16MB is only 1%. However, if the flush cycle is extended to one day, the proportion of the short-lived data rapidly decreases as the buffer size increases. When the buffer size is 64 MB, the short-lived data occupies 81%, but when the buffer size becomes 512 MB, it decreases to 58%. This is because a large number of host write requests are served by the buffer and only some of the write requests generate NAND write operations by the dirty buffer flush or by the dirty buffer replacement.



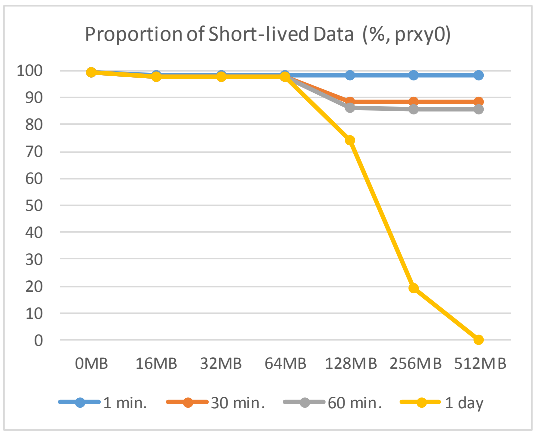
(a) hm0



(b) prn0



(c) proj0



(d) prxy0

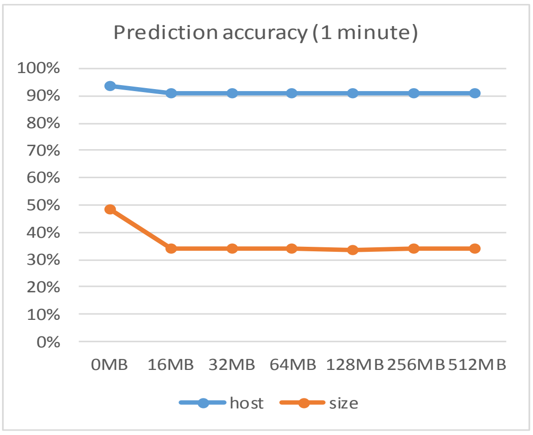
Fig. 1 Proportion of short-lived data

Prn0 trace shows a similar trend (Fig. 1 (b)). When the buffer is not used, about 81% of the data is short-lived data. When the flush cycle is 1 minute, about 72% of the data is short-lived regardless of the buffer size. That is, the proportion of data with a short lifetime is still high. As the flush cycle increases to 30 minutes, the proportion of short-lived data decreases to 61-65% depending on the buffer size. When the flush period is one day, the proportion of short-lived data decreases rapidly depending on the buffer size. That is, when the buffer size is 512 MB, only about 40% is short-lived data.

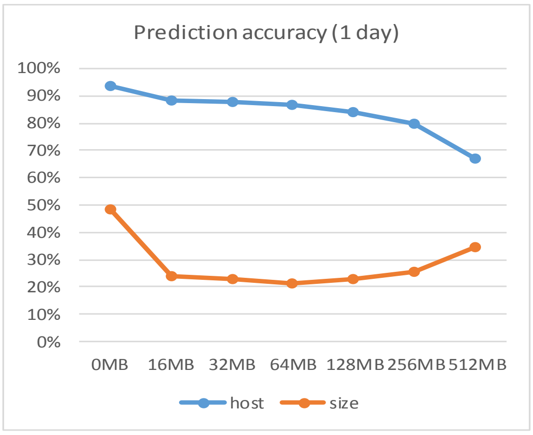
Fig. 1 (c) shows the result of proj0 trace. In proj0 trace, the proportion of short-lived data is very high. When the buffer is not used, about 99% of data is short-lived data. Even when the flush cycle is 1 minute, about 99% of the data is short-lived regardless of the buffer size. That is, the proportion of data with a short lifetime is still high. As the flush cycle increases to 30 minutes, the proportion of short-lived data decreases to 93-99% depending on the buffer size. When the flush cycle is long as one day, the decrease in the proportion of short-lived data increases depending on the buffer size. When the buffer size is 512 MB, about 73% of the data has a short lifetime.

Fig. 1 (d) shows the result of prxy0 trace. Also in prxy0 trace, the proportion of short-lived data is very high. When the buffer is not used, about 99% of data is short-lived data. Even when the flush cycle is 1 minute, about 98-99% of the data is short-lived. The effect of buffer size on the proportion of short lived data is small. When the flush cycle increases to 30 minutes, the proportion of short-lived data ranges from 89 to 98% depending on the buffer size. That is, the proportion of short-lived data is still very high, and as the buffer size increases, the proportion tends to decrease. However, when the flush cycle is one day, the proportion of short-lived data is drastically reduced depending on the buffer size. When the buffer is 16 MB, the short-lived data occupies 98%, but when the buffer size is 256 MB, it decreases to 19% and when the buffer size is 512 MB, it drops to 0%.

From the trace analysis results, we can draw the following common characteristics. First, in server traces, the proportion of short-lived data is very large. That is, the majority of data has a lifetime of less than one day. Second, if the internal buffer is present, the

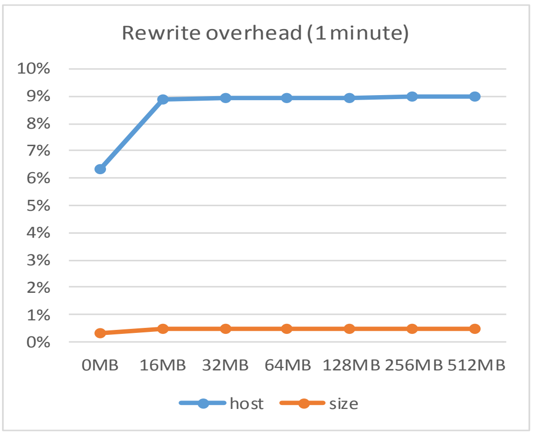


(a) 1 minute of flush cycle

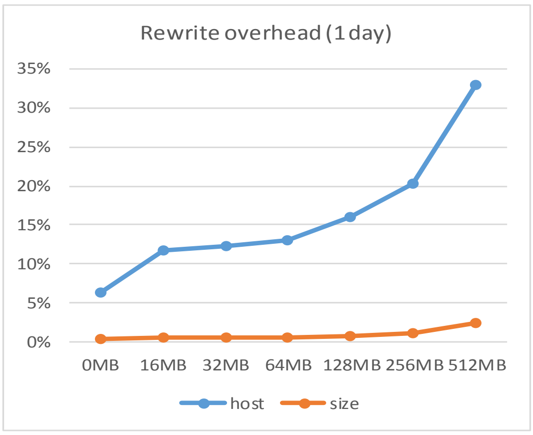


(b) 1 day of flush cycle

Fig. 2. Prediction accuracy in hm0 trace



(a) 1 minute of flush cycle



(b) 1 day of flush cycle

Fig. 3. Rewrite overhead due to incorrect prediction in hm0 trace

proportion of data with a short lifetime is somewhat reduced. However, the reduction varies depending on the buffer flush cycle. When the buffer flush cycle is short, such as 1 minute or 30 minutes, as the buffer size increases, the proportion of short-lived data decreases somewhat, but most data still have a short lifetime. However, if the buffer flush period is increased to one day, the proportion of short-lived data rapidly decreases as the buffer size increases. Depending on the trace, 73% of the data still has a short lifetime, while in other trace 0% of the data has a short lifetime.

Meanwhile, the reduced proportion of short-lived data is likely to hurt the accuracy of the lifetime prediction policies. Fig. 2 shows the prediction accuracy in hm0 trace. The host policy determines all host write requests as short-lived data, and the size policy determines the request size as 8 KB or less as short-lived data. The dirty buffer flush cycle is varied to 1 minute (fig. 2(a)) and to 1 day (fig. 2(b)). The result shows that when the internal buffer is used, the accuracy degrades in the both policies regardless of the flush cycle. When the buffers are flushed every minute, the accuracy of the host policy decreases from 93.7 (0MB buffer) to 91.0% (512MB buffer) and the accuracy of the size policy decrease from 48.7 (0MB buffer) to 33.9% (512MB buffer). The decrease of the accuracy is more conspicuous when the buffers are flushed every day. The accuracy of the host policy decreases to 67.1% (512MB buffer) and the accuracy of the size policy decrease to 34.6% (16MB buffer). Especially, the accuracy of the host policy drastically drops as the buffer size increases because the proportion of short-lived data greatly decreases.

As the prediction accuracy decreases, the incorrect prediction of long-lived data will increase, which results in the increase of the rewrite overhead. Fig. 3 shows the rewrite overhead in hm0 trace. This rewrite overhead is the ratio of the incorrect prediction of long-lived data for the total predictions. When the buffers are flushed every minute, the overhead increases from 6.3 (0MB buffer) to 9.0% (512MB buffer) in the host policy. In the size policy, the overhead increases from 0.3 (0MB buffer) to 0.5% (512MB buffer). The overhead increase is not serious.

However, when the buffers are flushed every day, the overhead of the host policy increases to 33.0% (512MB buffer), which is significant. In the size policy, the overhead increases to 2.5% (512MB buffer). The increase is more conspicuous in the host policy when the buffer size is large.

IV. CONCLUSIONS

The SSD internal buffer size and dirty buffer flush cycle had a significant effect on the lifetime of the data. When the buffer flush cycle was short, most data had a short lifetime, regardless of the buffer size. As a result, the accuracy of the existing lifetime prediction policies did not drop significantly. However, if the buffer flush period became one day, the proportion of short-lived data rapidly decreased as the buffer size increased. Even in prxy0 trace, 0% of the data had the short lifetime when the buffer was 512MB. As a result, the accuracy of the host policy dropped significantly, and the rewrite overhead increased drastically as the buffer size increased. Therefore, the policy of evaluating the data lifetime in SSD must be designed considering the buffer size and dirty buffer flush cycle.

The evaluation results indicate that if NVRAM (Non-volatile RAM) is used as the internal buffer of SSDs, the dirty buffer flush will not occur frequently and the proportion of short-lived data will be low, so identifying short-lived data and applying the shallow write to them is not likely to be effective. However, if DRAM is employed, the dirty buffer flushing is inevitable, and therefore, most data are likely to be short-lived and the shallow write will still be effective. As a future study, we plan to improve the performance of SSDs with DRAM by designing the lifetime prediction policy considering the dirty buffer flush cycle and by processing the short-lived data in the shallow write.

ACKNOWLEDGMENT

This study was supported by the Research Program funded by the SeoulTech (Seoul National University of Science and Technology).

REFERENCES

[1] R. Liu, C. Yang, and W. Wu, "*Optimizing NAND flash-based SSDs via retention relaxation*," in Proc. USENIX FAST, 2012.

[2] I. Shin, "*Applying fast shallow write to short-lived data in solid-state drives*," IEICE Electronics Express, vol. 15, pp. 1–9, 2018.

[3] M. Park, K. Lee, and I. Shin, "*Evaluating lifetime of server data based on trace analysis*," International Journal of Engineering Research and Technology, vol. 12, pp. 1441-1444, 2019.

[4] Y. Yao, X. Kong, J. Zhou, X. Xu, W. Feng, and Z. Liu, "*An advanced adaptive least recently used buffer management algorithm for SSD*," IEEE Access, vol. 7, pp. 33494–33505, 2019.

[5] I. Shin, "*Applying fast shallow write to short-lived data in solid-state drives*," Journal of KIIT, vol. 17, pp. 31–38, 2019.

[6] Y. Won, J. Jung, G. Choi, J. Oh, S. Son, J. Hwang, and S. Cho, "*Barrier-enabled IO stack for flash storage*," in Proc. USENIX FAST, 2018.

[7] A. Ban, "*Flash file system*," U.S. Patent 5 404 485, Apr. 4, 1995.

[8] Y. Pan, G. Dong, Q. Wu, and T. Zhang, "*Quasi-nonvolatile SSD: trading flash memory nonvolatility to improve storage system performance for enterprise applications*," in Proc. IEEE HPCA, 2012.

[9] D. Narayanan, A. Donnelly, and A. Rowstron, "*Write off-loading: practical power management for enterprise storage*," in Proc. USENIX FAST, 2008.