International Journal of Engineering Research and Technology. ISSN 0974-3154, Volume 12, Number 9 (2019), pp. 1441-1444

Evaluating Lifetime of Server Data based on Trace Analysis

Minhyun Park[1], Kyungah Lee[2], Ilhoon Shin[3*]
[1-3]Department of Electronic & IT Media Engineering,
Seoul National University of Science and Technology, South Korea.

[*]Corresponding author

ORCID: [(3)]0000-0003-2819-0398

Abstract

SSD (Solid state drives) performance can be improved by applying a shallow write to short-lived data. For this, the lifetime of the data should be accurately predicted, and different NAND write mode should be applied according to the data lifetime. A wrong prediction can result in degrading the performance and the lifetime of SSDs. This study analyses the factors that affect the lifetime of the data and presents policies that predict the lifetime of the data more accurately. The presented policies improves the accuracy by 2.2-89.1% compared to the existing policies by considering both write request size and previous lifetime of the data.

Keywords: Lifetime of data, server workload, SSD

I. INTRODUCTION

SSDs based on NAND flash memory have gained much attention over the past few years because they have better performance and consume less heat and energy compared to hard disks. SSDs are widely used not only in PCs but also in servers and data centres as the price of NAND flash memory is gradually lowered due to the development of semiconductor processing technology.

For server workload, most data is updated at very short intervals [1, 2]. This means that we do not need to retain the written value to NAND for a long time. Therefore, the SSD performance can be much improved by applying a shallow write, which is fast but do not guarantee a long retention time of data [2, 3].

However, if the prediction is wrong, the performance is seriously degraded because the shallowly written data should be re-written in a deep write to guarantee a long retention time. This also reduces the lifetime of SSDs. Therefore, it is important to accurately predict the lifetime of the data. For this, this study analyses the factors that perpetuate the lifetime of the data and presents more accurate policies than the existing ones.

II. BACKGROUND AND RELATED WORK

II.I NAND flash memory

NAND flash memory expresses values by the amount of electrons charged to each cell's floating gate. For example, in a single level cell (SLC), the state where the floating gate is empty indicate 1, and the state where the floating gate is fully changed indicates 0. In order to change the state of the floating gate, sufficient voltage should be applied.

As semiconductor processing technology advances, the physical size of each cell gradually decreases, and the maximum number of electrons that can be charged to the floating gate is also decreasing. This makes it difficult to distinguish between 0 and 1. In the case of a multiple level cell (MLC), it is more difficult to distinguish each value because one cell represents two bits. Furthermore, the oxide film surrounding the floating gate can be damaged by repeated charging and discharging, resulting in a natural leakage of electrons even though no voltage is applied. If the electrons are discharged too much, the state can be interpreted wrong and a bit error can occur, which is called retention error [2].

Modern NAND flash memory performs an ISPP (Incremental Step Pulse Programming) write operation to charge electrons by gradually increasing voltage to accurately control the amount of electrons to be charged [4]. As the charged electrons are precisely controlled with a low threshold voltage, the retention time is increased. However, due to repetitive charging operations with the low threshold voltage, the latency of the write operation increases. On the other hand, using a high threshold voltage can speed up the write operation, but it can shorten the retention time because the amount of the charged electrons are not precisely controlled.

Current SSDs are using the low threshold voltage to guarantee a sufficiently long retention time. However, in server workloads, a considerable amount of data have a short lifetime, which means that they do not need a long retention time. For the short-lived data, it is better to apply a shallow write, which uses the high threshold voltage, for a better performance. Only the long-lived data should be written in a deep write that uses the low threshold voltage.

II.II Related work

Server workloads have the characteristics of many data being updated at short intervals. Therefore, applying the shallow write to short-lived data can significantly improve the SSD performance. Several researches have proposed to exploit the shallow write for server workloads.

Liu and Yang assumed that all write requests are short-lived data and proposed to apply the shallow write for the write requests of hosts [2]. If the shallowly written data are not

modified for a certain period of time, they are re-written in the deep write mode to ensure a retention period, which results in restricting the performance improvement and decreasing the lifetime of SSDs.

To improve the accuracy of the prediction of data lifetime, Shin evaluated the data lifetime based on the size of the requested data [3]. A small sized request is evaluated to have a short lifetime and written in the shallow write. Conversely, a large sized request is evaluated to have a long lifetime and written in the deep write. This method is very accurate for the data with the short lifetime. However, for the data with the long lifetime, it turns out inaccurate.

III. PREDICTION OF DATA LIFETIME

III.I Trace analysis

Workload analysis was performed using Microsoft Research Centre (MSRC) traces [5], to find factors that affect data lifetime. First, the distribution of write request size is analysed and described in Table 1. From the result, the following can be seen: First, requests of 8KB or smaller accounts for 67.5 – 87.9% in the most traces excluding proj_0. That is, most write requests are small in size. Second, write requests larger than 32KB accounts for 5.3 – 11.7% in the most traces excluding proj_0. That is, for the most traces, large write requests are low in proportion. Third, there are traces that have the exceptional tendency. For example, in proj_0, 63.2% of the write requests exceed 32KB. Only 22.7 percent of the write requests have small size.

Table 2 shows the ratio of short-lived data to long-lived data for each trace. In classifying data according to their lifetime, data with a lifetime of less than one day are considered short-lived and data with a lifetime of longer than one day are considered long-lived. The results shows that for the most traces except prn_0 and prn_1, short-lived data account for 88.3 – 98.9%. That is, most of the data written on the server have a very short lifetime of less than one day. In prn_0 and prn_1 traces, the ratio of short-lived data is 74.8% and 67.0%, respectively, which is somewhat lower than those of other traces, but still, the ratio of short-lived data is high. Conclusively, server traces show that most data have a very short lifetime of less than one day, and thus it is not efficient to guarantee them a long retention time.

Table 1. Distribution of write request size

Trace	<= 8KB (%)	8KB < && <= 32KB (%)	> 32KB (%)
hm_0	76.7	13.7	9.6
mds_0	72.4	19.8	7.9
prn_0	79.5	8.9	11.7
prn_1	71.4	19.1	9.5
proj_0	22.7	14.1	63.2
prxy_0	87.9	6.8	5.3
stg_0	72.3	18.6	9
wdev_0	71.6	18.6	9.8
web_0	67.5	23.8	8.7

Table 2. Distribution of data lifetime

Trace	Short-lived data (%)	Long-lived data (%)
hm_0	88.3	11.7
mds_0	95.2	4.8
prn_0	74.8	25.2
prn_1	67.0	33.0
proj_0	98.9	1.1
prxy_0	98.8	1.2
stg_0	97.3	2.7
wdev_0	94.2	5.8
web_0	90.0	10.0

To find factors that affect the life of the data, we first analysed the correlation between the write request size and its lifetime. Fig. 1 shows the ratio of short-lived data to the size of the write request. The X-axis is the size of the write request, and the Y-axis illustrates the percentage of short-lived data for each write request size. The result shows that the smaller the write request size, the higher the short-lived data ratio. For example, in all the traces except prn_1, the ratio of short-lived data account for 95.5 – 100.0% when the request size is equal or less than 2KB. When the request size is greater than 16KB and less than 32KB, the ratio of short-lived data is 92.4 – 97.6%. However, if the size exceeds 32KB, the ratio of short-lived data is relatively low, showing a distribution of 48.7 to 98.9%. For the prn_1 trace, the ratio of short-lived data is relatively low. However, the smaller the request size, the higher the ratio of short-lived data can be found to be the same. The results suggest that when the size of the write request is small, for example, equal or less than 32KB, the data is likely to be short-lived. However, if the size of the write request exceeds 32KB, the correlation between the request size and the lifetime is not significant.

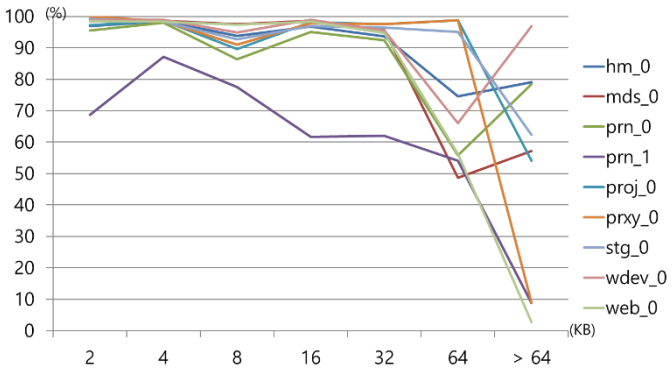

Fig. 1. Ratio of short-lived data for each write request size

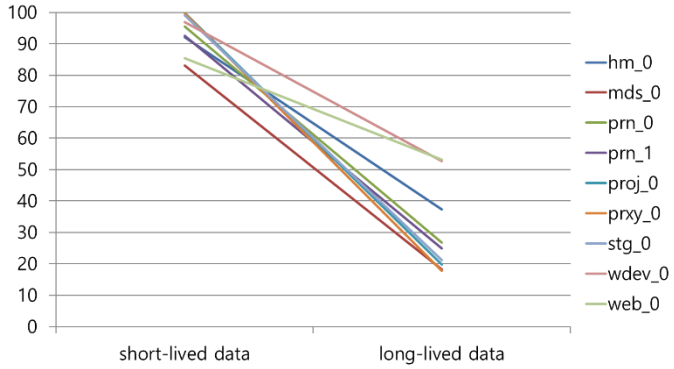

Fig. 2. Accuracy of predicting the lifetime of the next data using the lifetime of the previous data

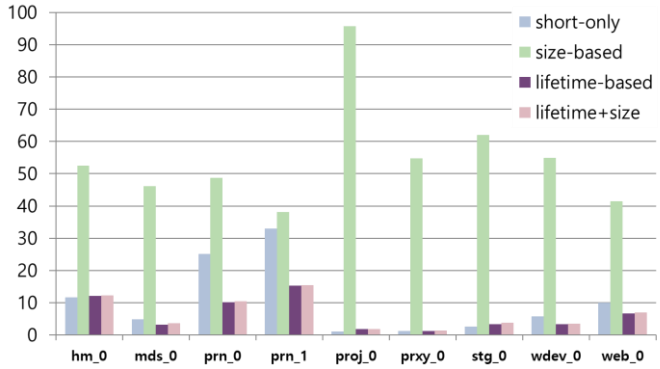

Fig. 3. Error rate of the lifetime prediction policies

To find another factor that affects data lifetime, we analysed whether the lifetime of the previous data written in the same sector affects the lifetime of the current data. Fig. 2 shows the accuracy of predicting the lifetime of the next data using the lifetime of the previous data. The probability that the next data's lifetime would be short is 83.1% to 99.9% for each trace when the previous data's lifetime is short. However, when the lifetime of the previous data is longer, the probability of the next data's lifetime being longer is 17.8 percent to 53.2 percent, with lower predictive accuracy. In other words, it is reasonable to predict that the lifetime of the next data will be short if the previous data's lifetime is short, but if the previous data has a long lifetime, it is not reasonable to predict the lifetime of the next data with the previous data's lifetime.

III.II Lifetime prediction policies and their accuracy

To accurately predict the lifetime of the data, two policies are designed: First, the lifetime-based policy predicts the lifetime of the data using the lifetime of the previously written data. If the data written in the same sector has a short lifetime, the current data is predicted to have the short lifetime, and if the previous data has a long lifetime, the current written data is predicted to have a long lifetime. Since the firstly written data does not have the lifetime of the previous data, the request size is used to estimate the lifetime. Equal or less than 32KB of data is expected to be short-lived, otherwise long-lived.

Second, the lifetime+size policy uses both the lifetime of the previous data and the size of the write request. If the data written in the same sector has a short lifetime, the current data is predicted to have the short lifetime. However, if the previous data's lifetime is long, then the lifetime of the current data is estimated by the request size. If the request size is equal or less than 32KB, the current data is expected to be short-lived, otherwise long-lived. For the firstly written data, the same estimation method with the lifetime-based policy is applied.

Fig. 3 shows the error rate of the policies. The short-only policy predicts all write requests as short-lived data [2], and the size-based policy predicts the lifetime by using only the size of the write request [3]. The result shows that the error rate of the size-based policy vary much depending on the trace, resulting in an error rate of about 5.5 – 90.9%. The short-only has an error of 1.1 – 32.9% depending on the trace. On the other hand, the error rates of the lifetime-based and lifetime+size policies proposed in this paper are 1.3 – 15.3% and 1.4 – 15.5%, respectively, showing improved accuracy compared to the existing policies. The difference between the lifetime-based and the lifetime+size policies is not significant.

Meanwhile, in all the traces except prn_1, the size-based policy has a higher error rate than the short-only policy, but [3] shows that the size-based policy performs better than the short-only policy. This is due to the difference in the overhead that occurs when short-lived data is mis-predicted as long-lived data and the overhead when long-lived data is mis-predicted as short-lived data. In other words, mis-predicting short-lived data results in an overhead of the time difference between the shallow write and the deep write because the data that can be written in the shallow write is actually written in the deep write. However, if long-lived data is mis-predicted, the data written in the shallow write should be re- written in the deep write later to ensure the retention period. Therefore, the number of NAND write operations and subsequently the number of block erase operations increase, resulting in degrading the performance more. Therefore, it is important to minimize the errors in predicting long-lived data incorrectly as short-lived data.

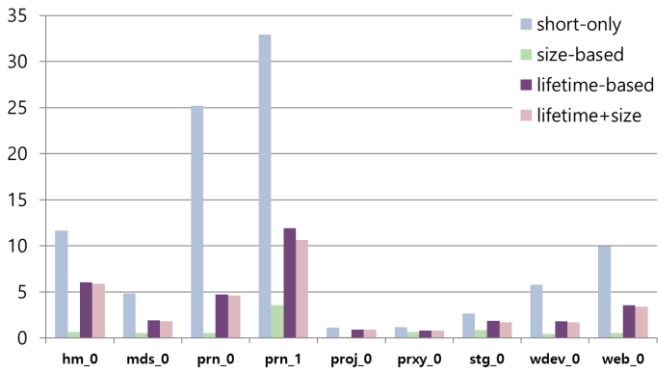

Fig. 4. Error rate for long-lived data

Fig. 4 shows the error rate at which long-lived data are incorrectly predicted as short-lived data. The result shows that the size-based policy has the lowest error rate. The error rate of the size-based policy ranges 0.2 – 9.6%. The error rate of the short-only policy is the highest and ranges 1.1 – 32.9%. The error rate of the life-based and the life-time+size policies is 0.8 – 11.9 % and 0.8 – 10.7%, respectively, which is higher than the size-based policy. Even though the overall error rate of However, the overall error rate of the life-based and the life-time+size policies is significantly lower than the size-based policy, the error rate at which long-lived data are incorrectly predicted is higher, which is the future research target.

IV. CONCLUSION

In this study, through the trace analysis, we extracted factors that affect the lifetime of the data and designed policies to predict the lifetime of the data based on the trace analysis result. The proposed policies showed improved the accuracy of the lifetime prediction by 2.2 − 89.1% compared to the existing policies. However, the error rate of misjudging long-lived data as short-lived data was higher than the size-based policy. Therefore, we plan to extract additional factors affecting data lifetime and improve the accuracy as a future work.

ACKNOWLEDGEMENTS

This work was supported by Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Education, Science and Technology (NRF-2016R1D1A1A09918559).

REFERENCES

[1] G. Soundararajan, V.Prabhakaran, M. Balakrishnan, and T. wobber, “Extending SSD lifetimes with disk-based write caches”, USENIX FAST Conference, 2010.

[2] R. Liu, C. Yang, and W. Wu, “Optimizing NAND flash-based SSDs via retention relaxation”, USENIX FAST Conference, 2012.

[3] I. Shin, "Applying fast shallow write to short-lived data in solid-state drives", IEICE Electronics Express, 2018:15(13):1–9.

[4] Y. Pan, G. Dong, Q. Wu, and T. Zhang, “Quasi-nonvolatile SSD: trading flash memory nonvolatility to improve storage system performance for enterprise applications”, IEEE HPCA Conference, 2012.

[5] D. Narayanan, A. Donnelly, and A. Rowstron, “Write off-loading: practical power management for enterprise storage”, USENIX FAST Conference, 2008.