Hybrid Solid State Drives for Improved Performance and Enhanced Lifetime

Yongseok Oh
University of Seoul
ysoh@uos.ac.kr

Eunjae Lee
University of Seoul
kckjn97@uos.ac.kr

Jongmoo Choi
Dankook University
choijm@dankook.ac.kr

Donghee Lee
University of Seoul
dhl_express@uos.ac.kr

Sam H. Noh
Hongik University
http://next.hongik.ac.kr

***Abstract*—As the market becomes more competitive, SSD manufacturers are moving from SLC (Single-Level Cell) to MLC (Multi-Level Cell) flash memory chips that store two bits per cell as building blocks for SSDs. Recently, TLC chips, which store three bits per cell, is being considered as a viable solution due to their low cost. However, performance and lifetime of TLC chips are considerably limited and thus, pure TLC-based SSDs may not be viable as a general storage device. In this paper, we propose a hybrid SSD solution, namely *HySSD*, where SLC and TLC chips are used together to form an SSD solution performing in par with SLC-based products. Based on an analytical model, we propose a near optimal data distribution scheme that distributes data among the SLC and TLC chips for a given workload such that performance or lifetime may be optimized. Experiments with two types of SSDs both based on DiskSim with SSD Extension show that the analytic model approach can dynamically adjust data distribution as workloads evolve to enhance performance or lifetime.**

I. Introduction

Solid State Drives (SSDs) are now popular in computer systems due to their superior performance. As the market becomes competitive, manufacturers are using MLC (Multi-Level Cell) flash memory chips that store two bits per cell in their SSDs. Recently, TLC flash memory chips that store three bits per cell are being considered as SSD components due to their low cost [1]. However, performance and endurance of TLC flash memory chips is considerably limited [2]. These performance and endurance limitations hinder developing SSDs based only on TLC chips.

One solution to this problem is to consider hybrid SSDs that integrate SLC and TLC chips or MLC and TLC chips into an SSD [3][4][5]. Ideally, such hybrid SSDs would hold the vast majority of the capacity in TLC chips allowing the cost to be comparable to a pure TLC SSD, while providing performance and endurance of the superior chips. For a hybrid SSD to provide high performance and endurance comparable to pure SLC or MLC SSDs, a sufficient amount of hot data should be retained in SLC/MLC chips such that a significant portion of I/O requests are handled in SLC/MLC chips. Hence, the key issue in designing a hybrid SSD is to find the optimal distribution of data among the SLC/MLC and TLC components that comprise a hybrid SSD. Also, as data deployment may differ for optimal performance and for optimal endurance, being able to dynamically balance performance and endurance during the life of the hybrid SSD would be even more beneficial.

In this paper, we propose a hybrid SSD solution, namely *HySSD*, that integrates SLC/MLC and TLC chips. Unlike previous approaches that have considered hybrid SSDs, we use an analytic model to decide the optimal deployment of data to SLC/MLC and TLC chips for a given workload. Furthermore, the analytic model that we propose can also be used to improve the lifetime of *HySSD* instead of performance. Through experiments with realistic workload traces, we show that the analytic model that we derive finds near-optimal deployment of data for performance and that we can significantly improve lifetime of hybrid SSDs through small sacrifices in performance.

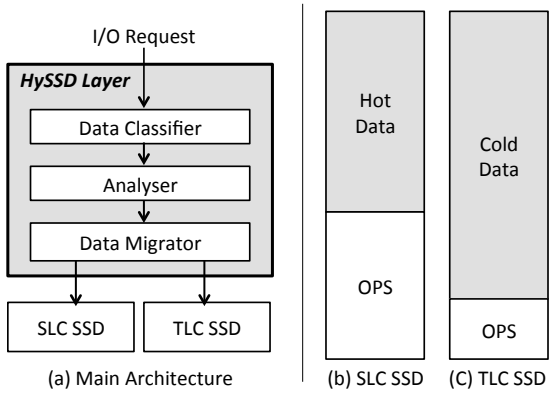
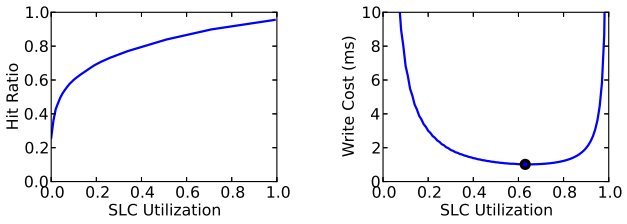


Fig. 1: Architecture of HySSD

II. Design of HySSD

In this section, we describe the design of a hybrid SSD. Figure 1 shows the overall architecture of *HySSD* that we envision. For brevity, we will concentrate only on the SLC and TLC hybrid SSD. However, our design and the model that we derive can be applied to any combination of SLC/MLC/TLC chips. We assume that the SLC chips and TLC chips are separately managed by their own page-mapping FTLs, and hence, we denote them as SLC SSD and TLC SSD in the figure.

On top of the SLC SSD and TLC SSD hardware, there sits the *HySSD* layer. The core function of the *HySSD* layer is to take in I/O requests (write requests, in particular) and decide which of the two SSDs should service these requests taking into consideration the overall performance and/or lifetime effect on the hybrid SSD. The software modules, namely, the *Data Classifier*, the *Analyser*, and the *Data Migrator* are the core components that are involved in making these decisions. The Analyser, in particular, is the module that employs the analytic model that we will derive. In the following, we give a detailed discussion of the Data Classifier, Analyser, and Data Migrator modules.



(a) Write hit ratio (b) Write cost

Fig. 2: (a) Write hit ratio curve and (b) write cost graph for the hit rate curve

A. Data Classifier

The function of the Data Classifier is to observe the workload pattern and classify the data into hot and cold data, and then, hand this information over to the Analyser. The distinction between hot and cold for the Data Classifier is quite simple. If the data can reside in SLC SSD it is hot; if not, then it is cold. Hence, there can only be B_{SLC} hot data blocks, where B_{SLC} is the maximum number of data blocks that can be stored in SLC SSD. Even though many sophisticated methods have been suggested, we simply use an LRU list that keeps the recency information to determine the hotness of a data block.

B. Analyser

In this section, we describe the role of the Analyser. In the process, we derive the analytic model that determines the optimal data sizes in SLC and TLC SSDs for a given workload. Given the B_{SLC} number of hot data blocks from the Data Classifier, the Analyser, in the end, determines how many of these blocks will indeed be kept in the SLC SSD, while all other data blocks are sent to the TLC SSD.

Optimizing for Performance: Though write performance between SLC and TLC SSDs differs significantly, read performance difference between the two is small. Therefore, rearranging hot and cold data between them for read requests provide only negligible benefits but hurts endurance of TLC SSDs as well as the overall performance in many cases. This is supported not only by the performance model that we describe in this section, but also by the experimental results. Hence, our study focuses on write requests rather than reads. Thus, our *HySSD* rearranges data between two SSDs to optimize write performance but does not forcibly exchange data between them for read performance.

There may be one situation where considering reads could affect performance. This is an outlier situation where the workload is significantly read oriented, at which rearranging data may possibly deliver performance benefits that exceed the penalty paid for rearranging the data. In this study, however, we do not consider this special case.

Let us now derive the write cost model of *HySSD*. The Analyser receives hot data information from the Data Classifier. The Data Classifier ignores the read requests and collects hot data information by observing only write requests. Based on this information, it generates the cumulative hit rate function for write requests, $H_{w\_SLC}(u_{SLC})$, which can be visualized as a curve shown in Figure 2(a). In the figure, the *x*-axis is the utilization of the SLC SSD denoted as u_{SLC}. If all B_{SLC} data blocks fill the SLC SSD to its maximum capacity, then u_{SLC} is 1; on the other extreme, if no data is stored, u_{SLC} is 0.

Note that this cumulative hit rate function can be regarded as the write hit rate when x amount of data are kept in SLC SSD. Thus, given u_{SLC}, $H_{w\_SLC}(u_{SLC})$ returns the hit rate at SLC SSD assuming that the SLC SSD is filled up starting from the hottest data block down to the block that makes SLC SSD utilization u_{SLC}. Naturally, then, since H_{w_SLC} is simply the write hit rate, all write requests not hit on the SLC SSD will miss to the TLC SSD with write miss rate $1-H_{w\_SLC}(u_{SLC})$.

Given this hit rate curve, we now derive the analytic model that the Analyser uses to calculate the optimal data sizes of the two SSDs. Let S_{SLC} and S_{TLC} denote the total size of flash memory chips in SLC and TLC SSDs, respectively, while D_{SLC} and D_{TLC} denote the size of data in SLC SSD and TLC SSD, respectively. Recall that over-provisioning space is required for garbage collection and hence, $D_{SLC} < S_{SLC}$ and $D_{TLC} < S_{TLC}$ always holds. Let D_{HYBRID} be the total data size stored in the hybrid SSD, then $D_{HYBRID} = D_{SLC} + D_{TLC}$. We can now formally define the utilization of SLC SSD and TLC SSD as $u_{SLC} = D_{SLC}/S_{SLC}$ and $u_{TLC} = D_{TLC}/S_{TLC}$, respectively. Then, directly using the derivation from Oh et al. [6], the page (data) write cost of SLC SSD, C_{PW_SLC}, and of TLC SSD, C_{PW_TLC} becomes as follows:

$$C_{PW\_SLC}(u_{SLC}) = \frac{C_{GC\_SLC}(u_{SLC})}{\lfloor (1 - U(u_{SLC})) \cdot N_{P\_SLC} \rfloor} + C_{PROG\_SLC} \quad (1)$$

$$C_{PW\_TLC}(u_{TLC}) = \frac{C_{GC\_TLC}(u_{TLC})}{\lfloor (1 - U(u_{TLC})) \cdot N_{P\_TLC} \rfloor} + C_{PROG\_TLC} \quad (2)$$

where C_{GC} is the garbage collection cost, U is the utilization translation function, N_P is the number of pages per a block, and C_{PROG} is the page program cost. (Detailed derivations are referred to [6].)

Then, we expect that, with rate $H_{w\_SLC}(u_{SLC})$, the hybrid SSD writes new data to SLC SSD and, with rate $1 - H_{w\_SLC}(u_{SLC})$, destages the least significant data from SLC SSD to TLC SSD and writes new data to the SLC SSD. Since the destage incurs a read from SLC SSD and a write to TLC SSD, the write cost of the hybrid SSD, $C_{PW\_HY}(u_{SLC})$, can be derived as follows.

$$\begin{aligned} C_{PW\_HY}(u_{SLC}) = {} & H_{w\_SLC}(u_{SLC}) \cdot C_{PW\_SLC}(u_{SLC}) \\ & + (1 - H_{w\_SLC}(u_{SLC})) \cdot \\ & (C_{PR\_SLC} + C_{PW\_TLC}(u_{TLC}) + C_{PW\_SLC}(u_{SLC})) \end{aligned}$$

where C_{PR_SLC} is the cost to read a page from the SLC SSD.

Now we consider the read requests. Remember that the Data Classifier provides only hot data information of write requests and the Analyser actively migrates data for write performance but is passive for read requests. In general, read hit rate at SLC SSD increases as more data are kept in the SLC SSD though they are hot for write requests. Let us assume that $H_{r\_SLC}(u_{SLC})$ returns the read hit rate at SLC SSD when the SLC SSD is filled with hot data for write requests that makes SLC SSD utilization be u_{SLC}. Then, with read hit rate $H_{r\_SLC}(u_{SLC})$, read requests are satisfied by the SLC SSD with cost C_{PR_SLC} and, with read hit rate $1 - H_{r\_SLC}(u_{SLC})$, read requests are satisfied by the TLC

SSD with cost C_{PR_TLC}, and thus, the overall read cost of *HySSD* can be calculated as follows.

$$C_{PR\_HY}(u_{SLC}) = H_{r\_SLC}(u_{SLC}) \cdot C_{PR\_SLC} + (1 - H_{r\_SLC}(u_{SLC})) \cdot C_{PR\_TLC}$$

Let us now assume that IO_R and IO_W are the rates of read and write requests, respectively. For example, if there are 40 read requests and 60 write requests out of 100 requests, then IO_R and IO_W are 0.4 and 0.6, respectively, and these numbers can be easily obtained in real systems. Then, we can calculate the overall access cost of the *HySSD* as follows.

$$C_{HYSSD}(u_{SLC}) = C_{PW\_HY}(u_{SLC}) \cdot IO_W + C_{PR\_HY}(u_{SLC}) \cdot IO_R \quad (3)$$

Our goal then boils down to finding u_{SLC} such that Eq. 3 is minimized. Let us take an example. Assume that the capacity of SLC and TLC SSDs, S_{SLC} and S_{TLC}, are 2GB and 8GB, respectively, and that a total of 9 GB of data is currently stored in the hybrid SSD. Note that 1 GB of capacity is over-provisioned space (OPS) and may be used during garbage collection. Assume also that, from the hot data information given by the Data Classifier, the Analyser finds the hit rate function to be as depicted in Figure 2(a). Then, using Eq. 3 with measured IO_R and IO_W values, the Analyser can calculate the overall hybrid SSD access cost according to u_{SLC} as shown in Figure 2(b).

Based on this result, the optimal u_{SLC}, that is, the utilization that minimizes the access cost is determined. For example, if the utilization is 0.63, the total data amount that should be placed in SLC SSD becomes $u_{SLC} \times S_{SLC}$, that is, 0.63×2GB $= 1.26$GB for this example. The rest of the data, that is, 7.74GB should be stored in the TLC SSD.

To determine the optimal sizes, the Analyser, which is the key component of *HySSD*, calculates the optimal u_{SLC} with an iterative algorithm. Starting from $u = 0$, the algorithm calculates the overall access cost of the *HySSD*, iterating as u is increased. Note that these calculations are only done periodically. In our implementation, we set the period to a logical time of N write requests.

Lifetime Management: In this section, we describe the lifetime management scheme of *HySSD*, for which, we make two assumptions. First, we assume that all flash memory blocks are evenly utilized via a wear-levelling scheme supported by the FTL. The use of such a simple wear-levelling model suffices as our goal is in deriving a general lifetime model for a hybrid SSD. We believe that typical wear-levelling schemes will be conformable within the framework of this general model. The other assumption regards the definition of lifetime. We will regard the hybrid SSD's lifetime expired when one of either SSDs inside exhausts its lifetime. Therefore, maximizing the lifetime of the hybrid SSD is to evenly utilize the lifetime of SSDs inside it.

Let us denote the maximum erase count of blocks in SLC and TLC flash memory chips to be E_{SLC}^{TOTAL} and E_{TLC}^{TOTAL}, respectively. Let E_{SLC}^{USED} and E_{TLC}^{USED} be the current average erase count of flash memory blocks in SLC and TLC SSDs whose values can be estimated by measuring the total amount of data written to it thus far. Then, E_{SLC}^{LEFT} and E_{TLC}^{LEFT}, the remaining number of erasures, becomes $E_{SLC}^{LEFT} = E_{SLC}^{TOTAL} - E_{SLC}^{USED}$ and $E_{TLC}^{LEFT} = E_{TLC}^{TOTAL} - E_{TLC}^{USED}$ for the SLC and TLC SSDs, respectively. Then, the relative remaining lifetime can be calculated as $L_{SLC} = \frac{E_{SLC}^{LEFT}}{E_{SLC}^{USED}}$ and $L_{TLC} = \frac{E_{TLC}^{LEFT}}{E_{TLC}^{USED}}$, respectively. Consequently, the remaining lifetime of a hybrid SSD can be derived as follows.

$$L_{HYSSD} = MIN(L_{SLC}, L_{TLC}) \quad (4)$$

To manage the lifetime of *HySSD*, first, the average erase counts of the flash blocks are estimated. In real products, the average erase count can be obtained through the S.M.A.R.T. command of SSDs. Otherwise, it can be calculated by measuring the total amount of data written to SSDs thus far. Then, the lifetime algorithm, which tunes the utilization of the SLC SSD (and hence, the utilization of the TLC SSD) to extend the remaining lifetime of *HySSD*, is executed. Specifically, if the SLC SSD has more lifetime to spare, then u_{SLC} is incremented to utilize the SLC SSD even more. Otherwise, u_{SLC} is decremented so that the TLC SSD is utilized more. This lifetime algorithm is periodically performed upon every N write requests.

C. Data Migrator

As *HySSD* periodically goes through the Data Classifier and the Analyser phases, information regarding which data blocks are hot and cold and how many blocks should be placed in SLC SSD and TLC SSD based on the measurement of choice is determined. The Data Migrator is the module that moves the data between the SLC and TLC SSDs according to the information provided by the Data Classifier and the Analyser. Some hot data residing in TLC may have to be moved to SLC, while some cold data residing in SLC may have to be moved to TLC.

Though data may be moved right away, this may incur unnecessary cost. Hence, to reduce migration cost, Data Migrator takes a passive approach. Specifically, it just waits until a write request to hot data in TLC SSD is summoned. As hot data has high probability to be requested, this passive approach will migrate hot data in TLC SSD without incurring extra cost. However, for cold data residing in SLC SSD, migration may not be so smooth as cold data may never be requested. Hence, we take a dual approach here; at first, we take the same passive approach as for hot data, but if the cold data is not written to even after some specified time, Data Migrator starts to force migration, but only in the background. This approach minimizes the migration effect on performance.

III. Performance Evaluation

In this section, we evaluate our *HySSD* design. For the workload, we use realistic workload traces that have been used in various other studies. The characteristics of the traces are summarized in Table I. For the performance evaluation, we implement *HySSD* based on CMU DiskSim with MSR SSD Extension. Our *HySSD* comprises the software modules described in Section II. The specific parameters for the simulation are given in Table II. For the experiments, we set up hybrid SSDs consisting of SLC, MLC, and TLC SSDs whose flash memory characteristics are listed in Table II.

TABLE I: Characteristics of I/O workload traces

Workload	Avg. Req. Size (KB)		Request Amount (GB)		Read Ratio
	Read	Write	Read	Write	
Financial [7]	5.73	7.2	6.76	28.16	0.19
Exchange [8]	9.81	12.56	37.43	41.34	0.48

TABLE II: Flash memory specifications

Description	Notation	SLC	MLC	TLC
Read Cost	C_{PR}	135us	175us	350us
Program Cost	C_{PROG}	350us	1400us	2500us
Copyback Cost	C_{CP}	385us	1375us	2450us
Erase Cost	C_E	1.5ms	3.8ms	3ms
# of pages per block	N_P	128	256	384
P/E Cycles	E_{TOTAL}	60,000	3,000	500
Page Unit Size	-	4KB	4KB	8KB

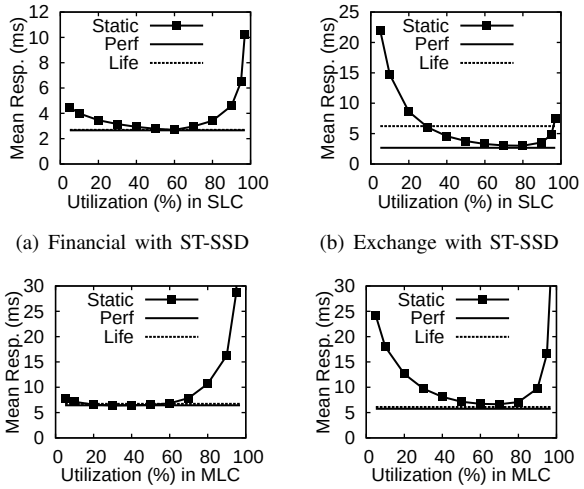
We present results for two hybrid SSDs. The first hybrid SSD denoted "ST-SSD" has 2GB SLC and 16GB TLC SSDs. The other hybrid SSD denoted "MT-SSD" has 2GB MLC and 16GB TLC SSDs. For a fair comparison, the hybrid SSDs are set to maintain a 2GB OPS size, while providing 16GBs of data capacity to the host.

For the results, we present the performance evaluation for three schemes. The results reported as "Static" is the scheme that statically sets the utilization of the superior SSD to a constant value. We consider a wide range of constant values ranging from 5% to 97%. We use these results to determine the best possible distribution for the particular workload. The other two are the dynamic schemes based on our analytical model. Since they are dynamic, the utilization of the superior SSD will vary according to the workload characteristics. Hence, only one value is reported for each of the two schemes. The first scheme denoted "Perf" is optimized for performance. The other scheme denoted "Life" additionally adjusts the performance optimized utilization to make more use of the "under-utilized" SSD so as to extend the overall lifetime of the hybrid SSD. The two schemes are realizations of the performance model and the lifetime management scheme described in Section II-B.

A. Response Time

Figure 3 shows the response times of the hybrid SSDs employing the "Perf", "Life", and "Static" schemes. The x-axis represents the utilization of the superior SSD for the "Static" scheme. The y-axis denotes the mean response time of I/O requests for all schemes. In contrast, the response times of "Perf" and "Life" are depicted as a horizontal line because each reports only one response time as it dynamically adjusts the utilizations of hybrid SSDs depending on the workload pattern.

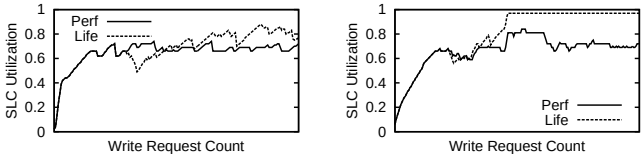
In the figure, the response time of "Static" scheme drops as utilization increases from 0, culminating at a minimal point at some utilization, and then increases again beyond this point. As the minimal point has been found after exploring all possible configurations of the "Static" scheme, this point can be regarded as the off-line optimal. In Figures 3(a)-(b), which shows the results for ST-SSD, the optimal points are at 60% for the Financial trace and 80% for the Exchange trace. As shown in Figures 3(c)-(d), the optimal points of MT-SSD are at 30% for the Financial trace and 70% for the Exchange trace. From



(a) Financial with ST-SSD (b) Exchange with ST-SSD

(c) Financial with MT-SSD (d) Exchange with MT-SSD

Fig. 3: Mean response time results of Hybrid SSDs



(a) Financial (b) Exchange

Fig. 4: Dynamic utilization adjustment of two dynamic schemes using ST-SSD

these results, we find that 1) there exists an optimal superior SSD utilization point where the response time is minimal and 2) that the optimal point is highly dependent on the workload pattern.

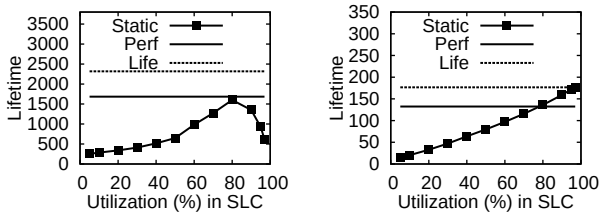
Now let us turn our attention to the performance of the "Perf" and "Life" schemes. All results show that the response time of "Perf" is close to the off-line optimal obtained from the "Static" scheme. This shows that the analytic model for performance is efficient in finding the optimal distribution of data for the given workload. On the other hand, "Life" shows slightly higher response time then "Perf" for all the traces. This is a natural result as "Life" adjusts utilization starting from the performance optimal point to elongate the lifetime of the hybrid SSD.

Figure 4 shows how the utilization of the superior SSD changes for "Perf" and "Life" as the workload changes for ST-SSD. These figures show that our scheme is adjusting the use of the hybrid SSDs according to the needs of the workload.

B. Lifetime

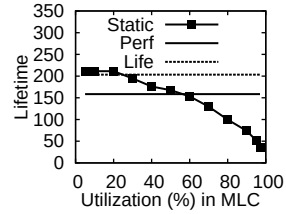
To compare the remaining lifetime of the hybrid SSD, we use the lifetime model, L_{HYSSD}, described in Section II-B. This metric represents the number of times the hybrid SSD may still be used based on the amount of data currently written. A larger value means that there is more life to live.

Figure 5 shows the lifetime results of the hybrid SSDs. In all the figures, the "Static" scheme shows a mountain shape result having left and right slopes. The left slope represents a situation where the superior SSD is less-utilized while the inferior SSD is being over-utilized, resulting in faster expiration of the hybrid SSD, and vice versa for the right slope. At the peak of the mountain, the two SSDs are being

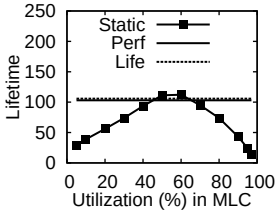


(a) Financial with ST-SSD

(b) Exchange with ST-SSD



(c) Financial with MT-SSD



(d) Exchange with MT-SSD

Fig. 5: Lifetime results of Hybrid SSDs

evenly utilized culminating in the longest lifetime of the hybrid SSD. Specifically, in the case of ST-SSD (Figures 5(a)-(b)), the optimal points are at 80% for the Financial trace and 97% for the Exchange trace. For MT-SSD (Figures 5(c)-(d)), we see that the optimal points are at 10% for the Financial trace and 60% for the Exchange trace. From the experimental results, we find that there exists a utilization point that maximizes the lifetime for hybrid SSDs and that this point varies according to the workload characteristics.

We now move on to the results of the "Perf" and "Life" schemes. As shown in Figure 5, "Perf" has a shorter lifetime than the best of the "Static" scheme, and this is not surprising as "Perf" focuses on performance, not lifetime. In contrast, "Life" notably enhances the lifetime reaching up to the best of the "Static" scheme for most cases.

It is interesting that "Life" can considerably enhance lifetime by slightly sacrificing performance as these results and the results of the previous section indicate. From these results, we can conclude that lifetime is more sensitive to the data distribution policy than performance and that a relatively large degree of lifetime can be enhanced by sacrificing a small degree of performance.

IV. Related Work

Besides SLC flash memory that stores one bit per cell, new technologies such as MLC and TLC flash memory have been developed to store multiple bits per cell. Compared to SLC flash memory chips, MLC chips are lower in cost, but also inferior in terms of performance and lifetime. TLC chips have even worse performance and endurance levels than MLC chips though it is the best (so far) in terms of density. To take advantage of the various merits of various flash memory chips, numerous hybrid approaches have been studied. Chang proposes using an SLC cache inside an MLC-based SSD [3]. Im and Shin propose storing hot data in the SLC region and cold data in the MLC region [4]. Similarly, Park et al. propose a hybrid approach using both SLC chips and MLC chips in an SSD [5]. In its basic idea of combining various flash memory chips, these studies are similar to our study. However, our study is unique in that we propose and make use of a sophisticated analytic model to determine the optimal placement of data to SLC/MLC and TLC chips for a given workload. Furthermore, in addition to optimal placement of data for performance, our analytic model can also be used to elongate the lifetime of the SSD. This added feature allows us to balance lifetime and performance of the SSD based on the needs of the customers.

V. Conclusions

A variety of SSDs, each with their strong and weak points in terms of performance, endurance, and price, are available for purchase. Which to employ for our system is a question that is answered, in general, in an ad-hoc manner. In this paper, we considered a hybrid SSD solution as yet another viable option. In so doing, we presented *HySSD*, a framework for making use of hybrid SSDs, that uses an analytic model to determine the optimal data deployment between two different SSDs for a given workload. The proposed analytic model can be used to optimize for either performance or lifetime depending on its need. We implement the analytic models of *HySSD* on DiskSim with SSD Extension and perform extensive experiments with realistic workloads. The results show that our analytic modeling approach can dynamically adjust data deployment as workloads evolves resulting in near optimal performance. We also show that by using this design, we can significantly extend the lifetime of hybrid SSDs by slightly sacrificing performance.

Acknowledgment

This research was supported in part by Seoul Creative Human Development Program funded by Seoul Metropolitan Government(No. HM120006), by the National Research Foundation of Korea(NRF) grant funded by the Korea government(MEST) (No. 2012R1A2A2A01045733), and by Basic Science Research Program through the National Research Foundation of Korea(NRF) funded by the Ministry of Education, Science and Technology(2010-0025282).

References

[1] Samsung Releases TLC NAND BASED 840 SSD, *http://www.anandtech.com/show/6329/samsung-releases-tlc-nand-based-840-ssd.*

[2] L. M. Grupp, J. D. Davis, and S. Swanson, "The Bleak Future of NAND Flash Memory," in *Proceedings of the 10th USENIX Conference on File and Storage Technologies (FAST)*, 2012.

[3] L.-P. Chang, "Hybrid Solid-State Disks: Combining Heterogeneous NAND Flash in Large SSDs," in *Proceedings of the 13th Asia South Pacific Design Automation Conference (ASP-DAC)*, pp. 428–433, 2008.

[4] S. Im and D. Shin, "ComboFTL: Improving Performance and Lifespan of MLC Flash Memory using SLC Flash Buffer," *Journal of Systems Architecture*, vol. 56, no. 12, pp. 641–653, 2010.

[5] J.-W. Park, S.-H. Park, C. C. Weems, and S.-D. Kim, "A Hybrid Flash Translation Layer Design for SLC-MLC Flash Memory Based Multibank Solid State Disk," *Microprocessors and Microsystems*, vol. 35, no. 1, pp. 48–59, 2011.

[6] Y. Oh, J. Choi, D. Lee, and S. H. Noh, "Caching Less for Better Performance: Balancing Cache Size and Update Cost of Flash Memory Cache in Hybrid Storage Systems," in *Proceedings of the 10th USENIX Conference on File and Storage Technologies (FAST)*, 2012.

[7] UMASS Trace Repository, *http://traces.cs.umass.edu*.

[8] S. Kavalanekar, B. Worthington, Q. Zhang, and V. Sharda, "Characterization of Storage Workload Traces from Production Windows Servers," in *Proceedings of the IEEE International Symposium on Workload Characterization (IISWC)*, pp. 119–128, 2008.