Automated Lookahead Data Migration in SSD-enabled Multi-tiered Storage Systems

Gong Zhang*, Lawrence Chiu†, Clem Dickey†, Ling Liu* Paul Muench†, Sangeetha Seshadri†,
*College of Computing
Georgia Institute of Technology, Atlanta, Georgia 30332
† IBM Almaden Research Center, San Jose, CA 95120

***Abstract*—The significant IO improvements of Solid State Disks (SSD) over traditional rotational hard disks makes it an attractive approach to integrate SSDs in tiered storage systems for performance enhancement. However, to integrate SSD into multi-tiered storage system effectively, automated data migration between SSD and HDD plays a critical role. In many real world application scenarios like banking and supermarket environments, workload and IO profile present interesting characteristics and also bear the constraint of workload deadline. How to fully release the power of data migration while guaranteeing the migration deadline is critical to maximizing the performance of SSD-enabled multi-tiered storage system. In this paper, we present an automated, deadline-aware, lookahead migration scheme to address the data migration challenge. We analyze the factors that may impact on the performance of lookahead migration efficiency and develop a greedy algorithm to adaptively determine the optimal lookahead window size to optimize the effectiveness of lookahead migration, aiming at improving overall system performance and resource utilization while meeting workload deadlines. We compare our lookahead migration approach with the basic migration model and validate the effectiveness and efficiency of our adaptive lookahead migration approach through a trace driven experimental study.**

I. Introduction

Recently, a number of storage systems supporting Flash devices (SSDs and memory) have appeared in the marketplace such as NetApp FAS3100 system [1] and IBM DS8000 [2]. In order to fully capitalize on the benefits of SSDs in a multi-tiered storage system with SSDs working as the fastest tier, it is important to identify the right subset of data that needs to be placed on this tier given the limited capacity of SSD tier due to high cost per gigabyte. Specifically, we want to maximize overall system performance by placing critical, IOPS-intensive and latency-sensitive data on the fast SSD tier through two-way automated data migration between SSDs and HDDs. By working with a variety of enterprise class storage applications, we observe that many block-level IO workloads exhibit certain time-dependent regularity in terms of access patterns and temperature of extents (hot or cold). For example, in banking applications, IO workloads for account access and credit verification are typically heavier during certain hours of a day. However, such patterns may change from day-time to night-time, from day to day, from weekdays to weekends or from working days to public holidays. Thus, block-level IO profiling is the first step for building an automated data migration system. The next big challenge is to devise strategies and methods of how to utilize the capability of storage virtualization and multi-tiered storage to perform effective and non-intrusive data migration in order to maximize the overall storage performance and resource utilization.

In this paper, we present an automated lookahead data migration scheme, called LAM, which aims to adaptively migrate data between different tiers to keep pace with the IO workload variations, to maximize the benefits of the fast but capacity-limited SSD tier, and to optimize the overall system performance in terms of response time and resource utilization, while limiting the impact of LAM on existing IO workloads.

Based on workload variations and temperature of block level IO access (e.g., hot or cold extents) learned through IO profiling, we predict shifts in hot-spots of block-level extents and proactively migrate those data extents whose temperature expected to rise in the next workload into the fast SSD tier during a lookahead period. A key challenge in the LAM design is to understand and tradeoff multiple factors that influence the optimal lookahead migration window. We have implemented a prototype of LAM in an operational enterprise storage system. Our experience shows that lookahead migration between fast tiers and slow tiers effectively improves the utilization of scarce resource like SSDs, provides faster average access speed with reduced average cost, and significantly enhances the overall response time of the system.

The main contributions of this paper are two folds. First, we describe the need and the impact of adaptive deadline aware data migration through observation and analysis of IO workload scenarios from real world storage system practice. By introducing the basic data migration model in an SSD enabled multi-tiered storage system, we study the characteristics and impacts of several factors, including IO profiles, IO block level bandwidth, and the capacity of SSD tier, on improving overall performance of the tiered storage systems. Second, we present a "lookahead migration" framework as an effective solution for performing deadline aware, automated data migration, by careful managing the performance impact of data migration on existing runtime application workloads and maximizing the gains of lookahead migration. A greedy algorithm is presented to illustrate the importance of determining a near optimal lookahead window length on the overall system performance and a number of important factors, such as block level IO bandwidth, the size of SSD tier, the workload characteristics,

978-1-4244-7153-9/10/$26.00

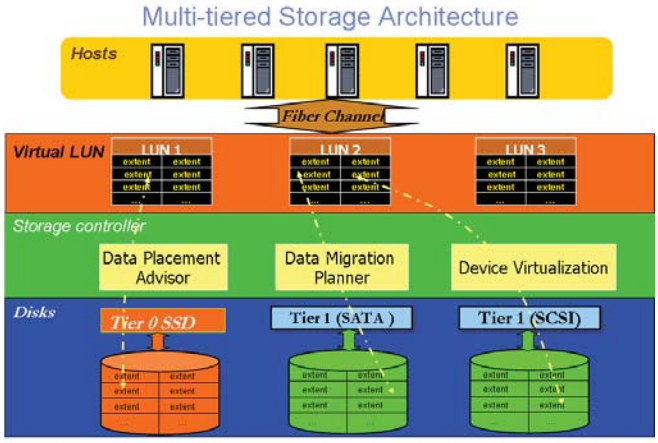

Fig. 1. Multitier Storage Architecture

and IO profiles. Our experiments are conducted using both the IO trace collected from benchmarks on a commercial enterprise storage system and the simulation over the real trace. The experimental study demonstrates that the adaptive lookahead migration scheme not only enhances the overall storage system performance but also provides significantly better IO performance as compared to both basic data migration and constant lookahead migration strategies.

II. LAM Overview

A. Multi-tiered Storage Architecture

Figure 1 gives a sketch of an SSD-based multi-tiered storage system architecture, in which SSD provides the high speed IO access as tier 0, and SATA disks and SCSI disks work as tier 1 to provide mass storage. Applications supported by the storage system operate the data in the unit of "logical extent" at storage virtualization level. Each logical extent is mapped to the physical extents contained at each tier in the storage hierarchy. In a nutshell, the multi-tiered storage provides a pool of physical extents of fixed size and applications organize their data in the form of logical extents of fixed size. It is the task of the storage controller to manage the mapping of logical extents in virtual disks to physical extents on physical disks. As we move data downward from tier 0 to tier 1, both cost and access speed in IOPS drop accordingly. In contrast, moving data upward from tier 1 to tier 0 provides applications with higher access speed but also higher price in terms of per GB access cost. Clearly, automated lookahead data migration between multi-tiered hierarchical concoctions of different types of storage devices can offer great flexibility in operating and tuning the tradeoffs between access speed and cost, ultimately improving the overall performance of multi-tiered storage systems.

B. Motivation Scenario and Assumptions

Through our experiences working with a number of enterprise-class storage system applications in banking and retail store industries, we observe that many IO workloads exhibit time-dependent workflow patterns in terms of random IO reads, random IO writes, and batch IO reads. Furthermore, many applications have similar IO workloads as those in banking or retail stores in terms of transaction processing, write-ahead logging, and real-time data analytics. Another interesting observation is that such IO workloads typically alternate between periods of high activities and low activities. Furthermore, the workload characteristics may change from daytime to night time activities, or from weekday to weekend activities.

For example, in a commercial banking environment, the workloads during the business hours primarily focus on the transaction processing tasks, such as account updates, credit verification. These tasks typically create intense accesses to indices and some metadata out of the large banking data collection. However, during the non-business hours, the workload type is often switched into report generating style and the high access frequency data may include different indices or different access logs, and different sets of metadata. Such workload patterns alternate from one period to another. Thus, hot data extents move from time to time, demanding adaptive data migration strategies to make the hot extents, such as frequently accessed indices, transaction logs, available in the high speed SSD tier. Improving the access speed for these "hot data" is highly critical to the overall performance enhancement of the entire multi-tiered storage system. Furthermore, we run the Industry standard IO benchmark SPC1 and TPCE for a duration of time on a proprietary enterprise storage system respectively and confirm the stability of hot data band for a given workload during its running cycle. We obtain two interesting observations from these experiments. First, for the two different workloads, the hot extents bands are totally different. Second, there exists stability of hot data and cold data within each workload cycle for both SPC1 and TPCE workloads. Readers may refer to our technical report [3] for further detail on the stability phenomenon.

The high disparity of access speed between traditional HDDs and SSDs and the highly cyclic behaviors of IO workloads motivate us to proactively migrate the hot data, like indices, into SSD for each workload type and thus significantly improve the access speed for hot data. To address the data migration problem that is constrained by the workload deadline, we identify the following set of assumptions:

1) The dominating IO workloads often exhibit cyclic behavior such that one type of workloads ends and another type of workloads start to dominate in terms of IO access density.
2) The data access patterns, especially the hot data, can be identified by analyzing historical access information.
3) Data migration for each specific workload pattern may be bounded by certain deadline. For example, in a banking environment, data migration scheduled to speed up the customer transactions during the business hours should finish before the desired migration deadline, which is before the non-business hour workload (check

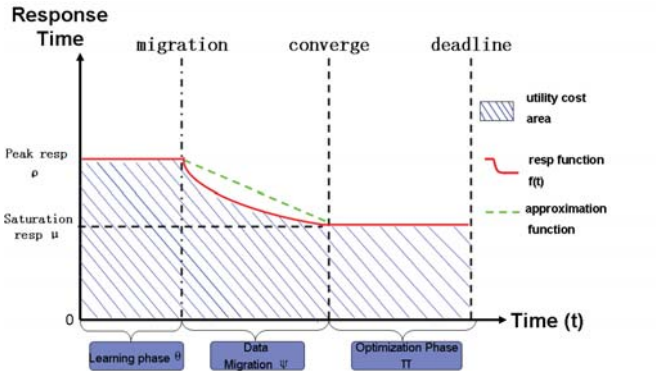

Fig. 2. Migration Computation Model

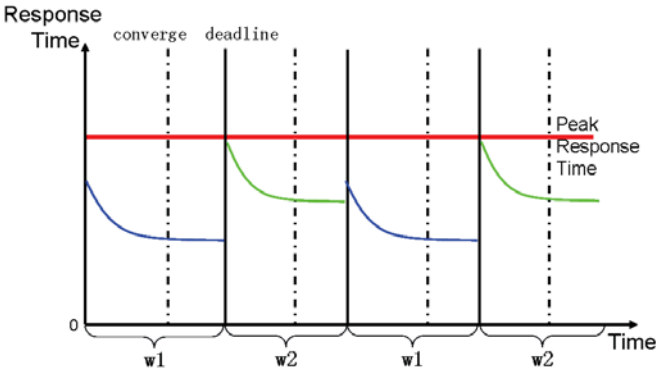

Fig. 3. Peak response time unreduced in the case without lookahead migration

processing and report generation) starts.

Based on the above assumptions, we develop an adaptive deadline-aware lookahead data migration framework − LAM, aiming at improving overall system performance and resource utilization through adaptively migrating hot data between fast storage tier and slow storage tier while meeting the deadline requirements.

C. Storage Utility Cost

In multi-tiered storage systems, response time is the major performance optimization goal for most of the users. Thus, the storage utility cost for a given time period is defined based on the response time over that period. We first define the response time function for a given time period as follows:

Definition 1. A function recording the response time during a time period from $t1$ to $t2$ for the multi-tiered storage system is called *"response time function"* and is denoted as $f(t), t \in [t1, t2]$.

Apparently, the mean or average response time over the system running period is a critical indicator of the system performance. The smaller the cumulative response time level is, the better the system performance. We denote this concept as the *"system utility cost"*, which is defined as follows:

Definition 2. The system utility cost, denoted by U, is the summation of the response time of the multi-tiered storage system at each time t over the entire time period of $[t1, t2]$ and $t \in [t1, t2]$. That is, $U_{t_1,t_2}(t) = \int_{t1}^{t2} f(t)dt$.

For example, the solid curve in Figure 2 shows a response time function $f(t)$ and the shaded area highlights the utility cost. Our goal is to minimize the system utility cost while guaranteeing the migration deadline. If migration is not employed, then the system response time is retained at its unoptimized level, which is slower compared with the improved response time through data migration.

D. Basic Migration Model

Basically, the migration process can be divided into three phases based on its impact on the overall response time of the system.

(1) Workload learning phase: In this phase, the heat of the extents are collected and highly frequently accessed data areas are identified and aggregated into the heat data list. It is important to note that very limited system resource is allocated to this process such that the impacts of the heat collection activities are reduced to the least extent. The statistics collected in this phase update the heat information of the data extents, which guide the actual data migration activities in the subsequent phases.

(2) Data migration: Data migration starts from the hottest data such that the migration sequence corresponds to the heat order of data extents. One important observation from this phase is that the migration impacts on the response time observes the law of "diminishing marginal returns". For the same volume of data migrated, initially response time is reduced at a more rapid rate and later the reduction speed is gradually slackened, and ultimately further data migration causes very little reduction in response time, namely a "convergence point" is reached. This is rooted in the ordered manner in which data is migrated, i.e., migration is employed in the descending order of heats.

(3) Optimization Phase: After the migration phase reaches the "convergence point", the system enters the "optimization phase", during which response time of runtime transactions maintains at the stable minimum level until the existing workload reaches its deadline and finishes its running period.

Figure 2 illustrates the three phases that a workload will experience in our basic migration model.

III. Automated Lookahead Data Migration

By utilizing the data heat stability we discussed earlier, one can turn on the learning phase in the beginning workload cycles. Once the data heat is stabilized, the learning phase can be reduced for the subsequent workload cycles by reusing the IO profile information learned at the beginning rounds of workload cycles. The reduction of learning cycles cuts off the redundant learning cost and enables the system to enter the optimization phase earlier than the basic migration model. Figure 3 displays the continuous workload cycles for two workloads in which the learning phase is reduced. Although this improvement reduces the redundant learning

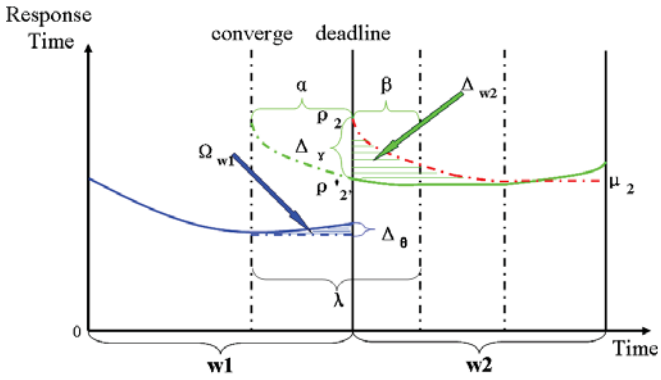

Fig. 4. Lookahead migration

phase, it offers no optimization after the workload reaches its convergence point. Thus, an important question is can we optimize the data migration performance after the convergence point is achieved and how should it be done. In this section, we propose an *automated lookhead migration* framework as an effective approach to optimize the post-convergence data migration stage.

A. Lookahead Migration

We base the learning phase reduced data migration model to illustrate lookahead migration in the context of Figure 4. The blue solid curve in $w1$ workload time and the green solid curve in $w2$ workload time represent the evolving process of response time of $w1$ and $w2$ respectively. The dash green line and the solid green line in $w2$ workload time together shows the effects of migration on response time of workload $w2$ with a configurable lookahead time window of size α. More concretely, after workload $w1$ enters the convergence stage, the data migration for $w2$ starts with α as the lookahead time window, namely the migration starts α units of time ahead of the time point when the $w1$ workload cycle reaches its deadline. The main reason that the lookahead migration is beneficial in this case is due to the fact that further data migration for ongoing $w1$ workload no longer generates response time reduction. Thus, it is more cost-effective if we start migrating the hottest data extents of $w2$ into the faster SSD tier ahead of activation time of $w2$.

Given the price per GB difference between HDD and SSD, and SSD tier has limited capacity in the multi-tiered storage hierarchy, thus it is very likely when the lookahead migration for $w2$ workload is activated, SSD has already been filled with the data extents of $w1$. In this case, the lookahead migration for $w2$ workload involves replacing the relatively cold $w1$ extents with the hottest $w2$ extents. This swapping process may increase the response time of ongoing workload of $w1$ to some extent. In Fig. 4 we observe the slightly bending upward of the solid response time curve of $w1$ after the lookahead migration of $w2$ begins. It is interesting to note that the dashed green line in the $w2$ cycle indicates that the lookahead migration with α as the lookahead window length reduces the peak response time of $w2$ workload experienced by the client applications by the amount of Δ_γ. Also the time duration that it takes to reach the convergence point perceived by the clients is reduced from a large $\alpha + \beta$ in the case of no lookahead migration to a smaller β under the lookahead migration for workload $w2$.

In terms of resource utilization gain represented by the shaded green area, denoted by Δ_{w2} and the resource utilization loss represented by the shaded blue area, denoted by Ω_{w1}, although with the achieved system utility gain Δ_{w2} in $w2$, $w1$ experiences a small amount of system utility loss Ω_{w1}, as long as the gain by Δ_{w2} is greater than the loss of Ω_{w1}, namely the condition of $\Delta_{w2} > \Omega_{w1}$ holds, lookahead migration with window size α will be cost-effective in terms of system utilization. With proper choice of the time window size α, we can maximize the difference of Δ_{w2} and Ω_{w1}, because we trade the relative cold extents in the end of the hot extent list of workload $w1$ with the hottest extents in the top of the hot extent list of $w2$. In terms of peak response time, similar condition of the difference of Δ_γ and Δ_θ holds. One can theoretically prove that the gain of Δ_{w2} over the loss of Ω_{w1} can be maximized by a proper setting of α and the quality of the heat information of the data extents. Due to the space constraint, we omit the theoretical proof and the details of the optimization algorithms in this paper. Readers may refer to our technical report [3] for further detail.

B. Greedy Algorithm Based Deadline Aware Lookahead Data Migration Approach

A naive approach of setting appropriate lookahead length is to set up the lookahead window length as a system-supplied constant, uniform for all types of IO workloads and IO profiles. Although bearing the advantage of simplicity, this approach is prone to set up a lookahead window length that is either too long or too short. If the constant lookahead window length is set to be excessive long, the response time of ongoing workload may be degraded seriously because too few hot data extents from the ongoing workload $w1$ are left in SSD. Too short lookahead window length will refrain the lookahead migration from the opportunities to produce the intended objectives. Thus selecting a balanced lookahead length is critical to harness the full power of lookahead migration.

Based on this observation, we design a greedy algorithm to adaptively compute the optimal lookahead length based on the IO profiles produced in the workload learning phase. The greedy algorithm starts from the lookahead window length of small scale. It computes the resource utilization gains from migration of $w2$ hot extents from HDD to SSD, denoted by the area marked by Δ_{w2} in Figure 4, and the performance loss of workload $w1$ due to the corresponding number of relatively cold extents of $w1$ that have to be swapped out from the SSD, denoted as the area marked by Ω_{w1}. For lookahead length α_i, if the difference between the gain and the loss, i.e., lookahead utility $\Gamma(\alpha_i) = \Delta_{w2} - \Omega_{w1}$ is larger than the lookahead utility of the previous lookahead length, $\Gamma(\alpha_{i-1})$, then we continue to increase the lookahead length by one more time unit. The algorithm repeats until a particular lookahead length achieves

smaller lookahead utility that its previous lookahead utility, i.e., $\Gamma(\alpha_i) < \Gamma(\alpha_{i-1})$. The lookahead length obtained at $(i-1)$ is the near optimum lookahead window length in our migration model.

This greedy algorithm is guaranteed to find the near optimal value of the lookahead window length for a given set of workloads. As workload changes or new workloads are introduced, after a few workload cycles of learning, new near optimal lookahead window length is obtained by using this greedy algorithm. Furthermore, we build a theoretical model of the lookahead data migration and the model is able to compute the optimal lookahead window size based on the IO profiles collected as an further improvement over this greedy algorithm. For further details, please refer to our technical report [3].

IV. Experimental Evaluation

A. Experimental Setup

To examine the benefits of lookahead migration, we first collected IO trace from real enterprise storage controller and implemented the lookahead data migration scheme in a simulator that simulates the real enterprise storage system. We developed a trace driven simulator, which consists of a storage unit with dual SMP server processor complexes as controllers, sufficiently large memory with large size cache, multiple types of disk drives (including SATA, SCSI and SSD), multiple FICON or ESCON adapters connected by a high bandwidth interconnect, and management consoles. Its design is to optimize both exceptional performance and throughput in supporting critical workloads and around the clock service.

Our simulator simulates the IO processing functionalities of the enterprise storage systems driven by the IO pressure trace collected from running benchmark workloads on the real enterprise storage system. For the experiments in this paper, the simulator simulates a multi-tiered storage hierarchy consisting of SSD drives as tier 0 and SATA drives as tier 1 and the total physical capacity is up to hundreds of terabytes in terms of hundreds of disk drives. The simulator mainly provides a simulation framework to plug in different data migration components. The simulation framework allows flexibility such that the data migration component can be directly moved to the real enterprise storage system. For the experiments in this paper, we run the simulations on a server machine with 4 processors and 12 GB memory running Linux Fedora 10.

IO traces for the simulator were generated by running the Industry standard IO benchmark SPC1 and TPCE on the referred proprietary enterprise storage system for a duration of 24 hours. In all the following experiment, "Average response time" refers to the average of the cumulative response time of all the requests for a extent in the extents pool at a particular time point and it is in the unit of miliseconds.

B. The Impacts of SSD Size on Data Migration

The number of SSDs impacts the data migration directly because more SSDs can provide faster access for more extents. In this set of experiments, we fix the data migration bandwidth and change the SSD size to observe the impacts on response time on the two workloads. Figure 5 shows the impacts of SSD size on 24 hours TPCE trace when we fix the migration bandwidth to $4GB/5mins$. The solid line represents the baseline response time when data migration is not employed. As we increase the number of SSDs, the response time is further reduced. The figure shows that using 1 SSD enables the system to reach minimum response time around $3700ms$ and achieve 17.8% improvement over the baseline case where only HDD is used. Similarly, 2 SSDs further reduces the stable minimum response time to the scale of $1840ms$ and improves the response time to 58%. Using 3 SSDS continues to improve the response time by 62.3% over the baseline case. When the SSD capacity is large enough to hold the majority of the hot extents, the impacts of increasing SSD size on the response time reduction is saturated, exemplified by the fact that the gap in stable response time between 2 SSD and 3 SSDs is smaller than the gap between 1 SSD and 2 SSDs.

C. Migration Bandwidth and Optimal Lookahead Length

Fig.6 examines the lookahead utility achieved by lookahead migration on a TPCE-SPC1 conjunct workload scenario and it confirms that greedy algorithm is able to identify a near optimal lookahead length. For example, when bandwidth 3GB/5mins, 4 hours lookahead window achieves near optimal lookahead utility. Fig.7 examines peak response time on the second workload $w2$ under increasing migration bandwidth from 2GB/5mins to 8GB/5mins while varying the lookahead length from 1 hour to 16 hours. The dotted line represents the peak response time measurement over lookahead length when 2GB/5mins is set to be the migration bandwidth. As lookahead length is increased from 1 to 6, the peak response time is significantly reduced by as much as 58%. Further increase of lookahead length larger than 6 hours only produces limited reduction on the peak response time. The dash curve shows a similar trend in the scenario when we set the migration bandwidth to be 4GB/5mins. The peak response time is reduced continuously at a dramatic rate until the lookahead lengh reaches 4 hours. The solid line shows the setting of 8GB/5mins, in which the impacts of increasing lookahead length on peak response time reduction is also saturated around 4 hours, except that because of higher migration bandwidth, for the same amount of lookahead length increment, the peak response time reduction is larger and quicker than the scenario with 4GB/5mins or lower migration bandwidth.

D. SSD size on Fixed Lookahead Length

Fig.8 shows the response time reduction process when we use SSD size ranging from 1 to 4 SSDs. As the SSD size increases, the converged response time is further reduced. The larger the SSD size is, the lower the peak response time will be. As the size of SSD tier increases, more and more hot extents are migrated into the SSDs. Given that the majority of the hot extents are migrated into the SSDs, further increasing the SSD size does not help to further reduce the response time. This can be exemplified by the overlap of the dotted

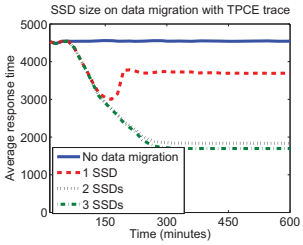

Fig. 5. Data migration with different SSD sizes with TPCE trace with 32 HDDs and 4GB/5mins migration bandwidth allocated

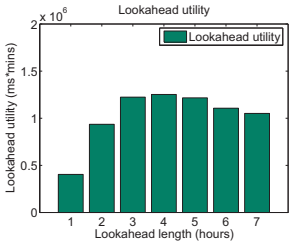

Fig. 6. Lookahead utility on incremental lookahead length over with bandwidth as 3GB/5mins for TPCE-SPC1 workload with 3 SSDs

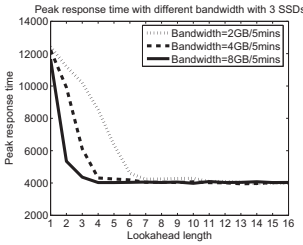

Fig. 7. Peak response time reduction on incremental lookahead length under different migration bandwidth in the second workload (SPC1) of TPCE-SPC1 trace with 3 SSDs

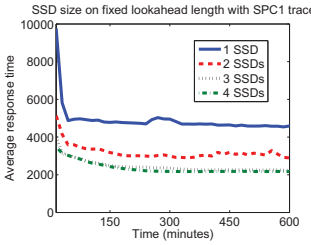

Fig. 8. SSD size on response time in fixed 4 hours lookahead length in SPC1 workload ($w2$) in TPCE-SPC1 with 32 HDDs and 4GB/5mins migration bandwidth allocated

line representing the 3 SSD scenario and the dash-dotted line representing the 4 SSDs scenario. Such overlap shows that although 4 SSDs has larger capacity than 3 SSD, 3 SSDs is large enough to accommodate the hot extents of the SPC1 workload, and thus any further increase of SSD size to 4 cannot lead to further reduction on response time.

V. Related Work and Conclusion

Recently two trends are presented in integrating SSDs into storage systems: on one hand, there have been some proposals on using SSD as one component of storage systems. On the other hand, designing a storage system which entirely comprises of SSDs is discussed in [4]. In this paper we promote the integration of SSDs as the high speed tier in multi-tiered storage systems. Further we argue that automated lookahead migration is critical for effective integration of SSDs into multi-tiered storage systems. In the literature, there are sereral migration techniques proposed. For example, [5] propose a storage system which evaluates the component's ability in supporting ongoing workload, and selects the candidates to migrate, aiming at satisfying the workload requirements. However, the migration scheme they proposed lacks of consideration on IO bandwidth, SSD size, and migration deadline and other constraints. QoSMig[6] proposes an adaptive admission control scheme to achieve the balance between client performance and migration objectives in the process of bulk data migration. However, none of them have investigated how to improve system performance and resource utilization through automated data migration in SSD-enabled multi-tiered storage systems.

In this paper, we have developed a lookahead migration scheme for efficiently integrating SSDs into the multi-tiered storage systems through deadline aware data migration. We present a greedy algorithm that finds the near optimal migration window size by adaptively tuning the lookahead window size and optimizing the lookahead impacts on improving overall system performance and resource utilization.

Our research continues along several directions. First, we argue that an important factor to be considered in any data migration mechanism is the choice of the right granularity of migration. The optimal granularity of migration is determined to a large extent by the actual application, data placement on the media, locality of accesses in the workload, workload characteristics (random vs. sequential, IO Size), acceptable metadata overhead and the total size of the system. At the granularity of an extent (1GB), our implementation required hundreds of MBs of memory to store metadata information for millions of extents. One of our ongoing research is on identification of the right granularity of migration based on the above characteristics. Another direction that is currently under investigation is effectively identifying hot data from IO profiles and evaluating the accuracy of our classification. In order to identify hot data based on workload IO profiles, a large amount of historical data needs to be collected and maintained to identify long-term trends in the workload. Furthermore, the sequence of when hot data extents become hot is another important factor for lookahead migration. We are investigating various summary statistics measures that can be utilized to address this concern.

VI. Acknowledgments

The work performed by Gong Zhang and Ling Liu are partially supported by grants from NSF NetSE and NSF CyberTrust, an IBM faculty award, an IBM SUR grant, and a grant from Intel Research Council.

References

[1] "NetApp FAS3100 System," http://www.netapp.com/us/products/storage-systems/fas3100/.

[2] "IBM DS8000," http://www-03.ibm.com/systems/storage/disk/ds8000/.

[3] G. Zhang, L. Liu *et al.*, "Automated lookahead data migration in SSD-enabled multi-tiered storage systems," *Technical Report*, 2010.

[4] V. Prabhakaran, T. L. Rodeheffer, and L. Zhou, "Transactional flash," *Proceedings of the 8th USENIX Symposium on Operating System Design and Implementation*, December 2008.

[5] E. Anderson, J. Hall, J. Hartline, M. Hobbs, A. R. Karlin, J. Saia, R. Swaminathan, and J. Wilkes, "An experimental study of data migration algorithms," 2001.

[6] K. Dasgupta, S. Ghosal, R. Jain, U. Sharma, and A. Verma, "Qosmig: Adaptive rate-controlled migration of bulk data in storage systems," *Data Engineering, International Conference on*, vol. 0, pp. 816–827, 2005.