Microprocessors and Microsystems 39 (2015) 167–180

Contents lists available at ScienceDirect

Microprocessors and Microsystems

journal homepage: www.elsevier.com/locate/micpro

Extending flash lifetime in secondary storage

Chengjun Wang [a], Sanjeev Baskiyar [b,*]

[a] Department of Computer Information Systems, Vermont Technical College, Randolph Center, VT, 05061, United States
[b] Department of Computer Science and Software Engineering, Auburn University, Auburn, AL 36849, United States

ARTICLE INFO

Article history:
Available online 9 March 2015

Keywords:
Disk cache
Flash lifetime
Secondary storage

ABSTRACT

Unlike magnetic disks, NAND flashes can be written a limited number of times. As flash memory densities increase and cell sizes shrink, further decreases in write endurance is expected. Although some mitigation is achieved by wear leveling, write endurance remains a concern for write intensive applications. In this research, we use a DRAM cache to filter write traffic to flash by coalescing and merging overwrites. To handle integrity of data upon power failure, we use a supercapacitor to provide short duration backup power during which DRAM data can be retired to flash memory. The effectiveness of such a mechanism is not obvious considering that a large file-system cache already exists which also merges overwrites. We investigated: (i) a DRAM and a flash disk cache combo within a magnetic disk controller and (ii) a DRAM only cache when flash is a full secondary storage. Our simulations show that using a medium sized DRAM cache, flash lifetime doubles with lazy updates compared to early update policy. Moreover, miss ratio and average response times decrease as well. With little effort, our technique can be extended to improve the usable life of other emerging non-volatile memories, such as PCM and MRAM.

1. Introduction

The performance gap between the processor and storage of computers is widening with approximately 60% and 10% annual improvement in the processor and hard disk drives (HDDs) respectively [1]. The trend is becoming more marked with the advent of multi-socket, multi-core processor architectures. The I/O performance, especially I/O operations per second (IOPS), has not caught up with the corresponding improvements in processor performance. The processor utilization stays low due to the need to wait for the data being fed from the storage system [2]. Therefore, storage performance is becoming the bottleneck of a computer system.

Fortunately, there are emerging memory technologies that try to bridge the growing performance gap, such as flash memory, Phase Change RAM (PCM), and Magnetic RAM (MRAM). Noticeable among them is flash memory, which has been the most widely used nonvolatile memory. There are two types of flash: NOR flash and NAND Flash. NOR flash is byte addressable while NAND flash is page addressable. In this paper, we are only concerned about NAND flash, which is referred to as flash for short hereafter. Not only has flash been widely used on portable devices as storage media, but also flash-based Solid State Drives (SSDs) are being installed into data centers.

SSDs can provide as much as 3300 write IOPS and 35,000 read IOPS, consuming 2.5 W of power, whereas even the best HDDs (15 K RPM drives) can only offer 300–400 IOPS while consuming 15–20 W of power [2]. In comparison, the processor can offer 1,000,000 IOPS. Performance in term of IOPS is critical for enterprise applications serving a large number of users, like web servers, email servers, cloud computing, and cloud storage. The common method used to close the gap is to deploy multiple HDDs working in parallel to support peak workloads. In order to meet the IOPS requirement, more HDDs are added, which results in environments that have underutilized storage (well below 50% of their useful storage capacity). The extra storage in turn will incur power and cooling waste.

However, flash can only be written a limited number of times, ranging from 10 K to 100 K depending on the type of flash used. Traditional solutions to limited lifetime of flash focused on the algorithms used within Flash Translation Layer (FTL), which spread the writes evenly across the medium. It is referred to as wear leveling, in which no single cell fails ahead of others. Although wear leveling mitigates the lifetime issue to some extent, it remains a concern for write intensive applications.

In this paper, we focus on:

- Using DRAM as a cache for flash memories rather than HDD as a cache unlike previous works. Since DRAM does not have read penalty unlike HDD, the performance needs investigation.

* Corresponding author.
E-mail addresses: cwang@vtc.edu (C. Wang), baskisa@auburn.edu (S. Baskiyar).

http://dx.doi.org/10.1016/j.micpro.2015.03.002
0141-9331/

Read penalty impacts response time and throughput. By introducing HDDs into SSDs, the device as a whole will lose its advantages in terms of acoustic levels, mechanical reliability, susceptibility to environmental factors, weight and size, and power consumption.

- Determining whether a small DRAM (15–60 MB) can effectively reduce write traffic. Although earlier works [3] have used relatively larger size (91–2,728 MB) HDD as a cache for SSDs, the effectiveness in using smaller size caches have not been investigated. Using a small size cache is important. Upon power failure, DRAM contents need to be retired to permanent store using a supercapacitor, whose size will be small for retiring the contents of a small DRAM. Thus, the super-capacitor can be practically implemented in a reasonable space.
- Comparing the retirement policy for entries of DRAM, i.e. early vs. lazy retirement, in terms of traffic, response time, miss ratios and DRAM sizes.
- Determining the effectiveness of employing DRAM in increasing lifetime of the flash cache used in HDD controllers. We see that with a medium sized DRAM cache, flash lifetime doubles with lazy updates compared to early updates. Moreover, miss ratio and average response times decrease as well.

The remainder of the paper is organized as follows: Section 2 presents the the motivation. Section 3 briefly reviews related work. In Section 4, we talk about the system architecture. Section 5 discusses the methodology we used. In Section, 6, we show the simulation results when flash acts as disk cache. In Section 7, we present the simulation results when flash is used as major store. In Section 8, we conclude the paper and finally, in Section 9, we address future works.

2. Motivation

2.1. Flash memory

As mentioned above, one of flash drawbacks is write endurance. The endurance issue stems from cell degradation caused by each burst of high voltage (10 V) across the cell. The problem shows no imminent sign of vanishing. As flash goes into Multi-Level Cell (MLC), write endurance becomes worse compared with Single Level Cell (SLC). For example, the write cycles of 2X MLCs drop to 10,000 from 100,000 (SLC). Furthermore, write endurance becomes worse as cells become smaller.

In order to mitigate the flash drawbacks, a Flash Translation Layer (FTL) has been proposed to manage how the flash resources are used [4]. FTL keeps mapping tables between logical and physical address spaces. When an update to a file is issued, FTL writes the file to a blank page and marks the old page as invalid. FTL updates the mapping tables accordingly. A technique called wear leveling attempts to evenly use entire memory cells.

In spite of all these efforts, endurance is still a concern for flash based storage systems [5]. The flash lifetime over I/Os per second is shown in Fig. 1 [6]. The lifetime drops as the number of I/Os increases. As Kim et al. [6] put it, "Although MTTFs for HDDs tend to be of the order of several decades, recent analysis has established that other factors (such as replacement with next, faster generation) imply a much shorter actual lifetime and hence we assume a nominal lifetime of 5 years in the enterprise." As we can see from the figure, when the number of I/Os exceeds approximately 50 IOPS, the lifetime of flash is less than 5 years. For example, the OpenMail workloads we used have 98 I/Os per second, which corresponds to a lifetime less than 2 years. Therefore, the endurance issue is one of the factors that would hinder the further applications of flash-based storage systems to a heavy write-intensive workload environment

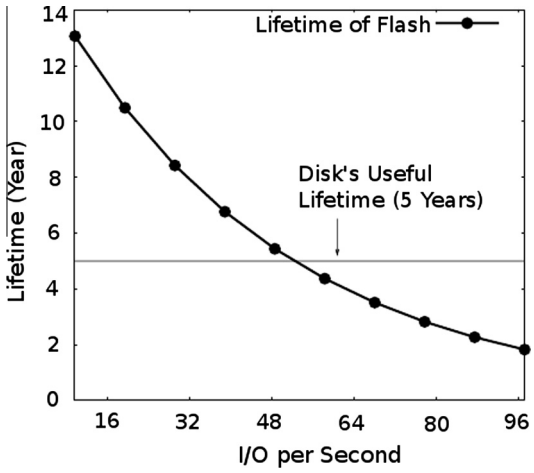

Fig. 1. Lifetime of flash.

like some embedded or enterprise systems, where the storage system needs to work 24/7 with heavy write traffic. Therefore, enhancing the flash endurance is demanded.

2.2. Flash trends

According to [7], flash trends can be expressed as: bigger, faster, and cheaper, but not better. Bigger: Flash component densities are doubling at a rate greater than Moore's Law. Faster: A single operation is needed for programming an entire page or erasing an entire block. Cheaper: As the densities are going up, price is going down. Not better: "Better" means more reliable in terms of endurance and data retention. Data retention is defined as the length of time a charge remains on the floating gate after the last program. The cells are worn out over program/erasure and make it more difficult to keep the electrons in place. Therefore, there is an inverse relationship between endurance and data retention—the more program/erasure, the shorter the data retention. As NAND manufactures are struggling for lower cost per bit, they keep sacrificing endurance and data retention. The most renowned trade-off is in MLC vs. SLC, in which 2:1 or 3:1 cost benefit is obtained through 10:1 reduction in rated endurance.

According to Hutchby and Garner [8], Flash has been classified as "mature nonvolatile memory". Most importantly, flash will not disappear in the short run [9]. Although there are emerging memories, such as PCM (PCRAM), and MRAM, they have not reached the maturity level to replace flash. Flash will coexist with the emerging memories for 5–10 years. Therefore, the lifetime issue of flash deserves the efforts of research.

2.3. Limitations of existing solutions

There are two basic methods to solve or mitigate the short flash lifetime (1) efficiently use cells, (2) reduce the write traffic to flash. Most research uses the first method, e.g., wear leveling. Since flash lifetime is directly related to cycles of program/erase, reducing write traffic to flash will extend the flash lifetime. However, little research has employed the second method. We did find Soundararajan et al.'s paper [3] using the second method. However, they use disk-based write cache instead of DRAM cache to save write traffic. Their research was based on a much larger cache size (90–2728 MB). The cache size of 15–60 MB was not investigated. Therefore, their conclusions do not apply to a DRAM cache. In addition, they used a log file, which results in read penalty and affects performance and response time. By using HDDs in SSDs, SSDs will lose their benefits in power consumption, weight