Contents lists available at ScienceDirect

Future Generation Computer Systems

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

DWARM: A wear-aware memory management scheme for in-memory file systems[☆]

Lin Wu[a], Qingfeng Zhuge[b,*], Edwin Hsing-Mean Sha[a,b], Xianzhang Chen[a], Linfeng Cheng[a]

[a] *Department of Computer Science, Chongqing University, Chongqing 400044, China*
[b] *School of Computer Science and Software Engineering, East China Normal University, Shanghai, China, 200241, China*

HIGHLIGHTS

- We reveal the problem of "hot spots" when updating data structures stored on NVM of in-memory file systems. Based on our experimental results, we find that the memory management schemes can potentially cause significant skewness in terms of writes to each data page.
- We propose an effective method to record the number of writes to NVM pages. We associate each page with a *write counter* and update the counter in file write path. All the counters are stored in a fixed area after super block of file system and accessed as an array by page frame number. We illustrate that the pages used for storing write counters cannot wear out with a good free page management scheme.
- We present a novel wear-ware memory management scheme for in-memory file systems, called *dynamic wear-aware range management*. This scheme divides the NVM pages into different wear ranges according to the write counters. Wear-leveling is achieved by selecting pages that have received the less writes for allocation requests.
- We propose *Adaptive Wear Range Determination Algorithm* to adjust the wear ranges dynamically.
- We propose an index structure to fast locate the appropriate wear ranges. The index is kept in DRAM to ensure high performance for allocation/deallocation.

ARTICLE INFO

Article history:
Received 30 September 2017
Received in revised form 19 January 2018
Accepted 27 February 2018
Available online 8 May 2018

Keywords:
In-memory file system
Wear leveling
Memory management
Index

ABSTRACT

Emerging non-volatile memories (NVMs) are promised to revolutionize storage systems by providing fast, persistent data accesses on memory bus. A hybrid NVM/DRAM architecture that combines faster, volatile DRAM with slightly slower, denser NVM can harness the characteristics of both technologies. In order to fully take advantage of NVM, state-of-the-art in-memory file systems are designed to provide high performance and strong consistency guarantees. However, the free space management schemes of existing in-memory file systems can easily cause "hot spots" when updating data structures on NVM, leading to significant skewness in terms of writes to each data page.

In this paper, we propose *dynamic wear-aware range management (DWARM)* scheme, a novel free space management technique for in-memory file systems. This scheme achieves wear-leveling with high performance for allocation/deallocation. The essential idea is to allocate less-written pages for each allocation request. Specifically, this scheme works by associating a *write counter* with each data page and updating the counters in the file write path. We build an "index" structure to fast locate the pages that have received less writes. The index divides the NVM pages into different subranges according to the write counters. Allocation always starts from the minimal subrange. Also, we propose *Adaptive Wear Range Determination Algorithm* to adjust the wear ranges dynamically. To accelerate lookup, we keep the index in DRAM and avoid the overhead of strong consistency by rebuilding the index in case of system failure. Experimental results show that this scheme can provide 4.9× to 158.1× wear-leveling improvement compared to the state-of-the-art memory management schemes. For application workloads, the *DWARM* strategy can improve the lifetime of NVM by up to 125×, 39×, and 25×, compared with the standard memory management schemes of PMFS, NOVA and SIMFS.

☆ This work is partially supported by National 863 Program, China [Grant No. 2015AA015304], NSFC, China [Grant No. 61472052].

* Corresponding author.
E-mail addresses: wulin_cqu@cqu.edu.cn (L. Wu), qfzhuge@gmail.com (Q. Zhuge), edwinsha@gmail.com (E.H.-M. Sha), xzchen109@gmail.com (X. Chen), linfeng_cqu@cqu.edu.cn (L. Cheng).

https://doi.org/10.1016/j.future.2018.02.038
0167-739X/© 2018 Published by Elsevier B.V.

1. Introduction

Emerging byte-addressable non-volatile memory (NVM) technologies, such as phase change memory, spin-torque transfer and Intel 3D XPoint [1] are promised to revolutionize I/O performance by providing fast, cheap and persistent storage-class memory systems. Since NVMs can be directly connected to the memory bus and accessed by CPU load/store instructions, many approaches have been proposed to integrate NVMs into computer systems [2–4]. Considering the wear-out and slower writes of NVM, a hybrid DRAM/NVM architecture that combines faster, volatile DRAM with slightly slower, denser NVM can harness the advantages of both technologies.

Hybrid DRAM/NVM storage systems provide lots of opportunities and challenges for system designers. New in-memory file systems are designed to take the most of NVM's high performance while providing strong consistency guarantees [5–8]. These file systems avoid software overhead of conventional storage stack optimized for traditional block devices. Specifically, all the file systems directly copy data between user buffer and NVM space without going through the generic block layer and OS page cache. SIMFS [5] further reduces software overhead by utilizing hardware Memory Management Unit (MMU) and continuous file virtual address space to access file data without software searching file metadata. NOVA [7] utilizes log and light-weight journal to provide data, metadata and *mmap* atomicity and keeps complex data structures in DRAM to accelerate lookup operations.

Unfortunately, one fundamental challenge is not addressed by state-of-the-art in-memory file systems: *how to avoid wear-out of NVM*. Since NVM cells can only endure a limited number of erases before it is worn-out, the lifetime of NVM devices can be threatened by uncontrolled write operations. This characteristic of NVM calls for usage patterns that can utilize and erase the NVM space evenly. In order to optimize the performance and endurance of NVM, hardware wear-leveling techniques have been proposed to make writes to NVM cells uniform [9]. However, hardware wear-leveling cannot provide a complete solution if frequent writes are issued to the same or neighboring NVM cells. More specifically, hardware achieves wear-leveling by transparently remapping the logical address space exposed to OS and physical NVM locations. Such operation can impose considerable overhead to deal with malicious write streams to a small set of locations [10,11].

As discussed in Section 2.2, the memory management scheme of existing in-memory file systems can easily cause "hot spots" when updating data structures of file system. We argue that a combined software–hardware approach is necessary to achieve wear-leveling while ensuring high performance and consistency guarantees.

In this paper, we propose a novel memory management scheme for in-memory file systems, called *dynamic wear-aware range management (DWARM)*. The goal of this scheme is to provide software wear-leveling while ensuring high performance for page allocation/deallocation. The essential idea is to choose pages that are less-written for each allocation request. Specifically, we first record the number of writes to each NVM page frame in the file write path. Then, we build an index structure to help locate the pages that have received relatively less writes. The index divides the NVM pages into different subranges, called *wear ranges*, according to the write counters of all the pages. The subranges are sorted by write counters and the minimal wear range contains pages that have received less writes. To enable fast lookup and updates, we keep the index structure in DRAM. For a page allocation request, we select a page from the minimal wear range. For a deallocation request, the freed page is put into the appropriate subrange according to the updated write counter. Besides, we propose *Adaptive Wear Range Determination Algorithm* to adjust the wear ranges dynamically as pages are allocated and freed. In addition, we eliminate extra overhead of ensuring strong consistency of the index by scanning the NVM free list to rebuild it in case of system crash.

We implement the *DWARM* scheme for different file systems in Linux kernel 4.4.4. Experimental results show the proposed memory management strategy can improve wear-leveling by up to 158.1×. For application workloads, the *DWARM* strategy can improve the lifetime of NVM by up to 125×, 39×, and 25×, compared with the standard memory management schemes of PMFS, NOVA and SIMFS.

In summary, this paper makes the following contributions:

- We reveal the problem of "hot spots" when updating data structures stored on NVM of in-memory file systems. Based on our experimental results, we find that the memory management schemes can potentially cause significant skewness in terms of writes to each data page.
- We propose an effective method to record the number of writes to NVM pages. We associate each page with a *write counter* and update the counter in the file write path. All the counters are stored in a fixed area after super block of file system and accessed as an array by page frame number. We illustrate that the pages used for storing write counters cannot wear out with a good free page management scheme.
- We present a novel wear-aware memory management scheme for in-memory file systems, called *dynamic wear-aware range management*. This scheme divides the NVM pages into different *wear ranges* according to the write counters. Wear-leveling is achieved by selecting pages that have received the less writes for the allocation requests.
- We propose *Adaptive Wear Range Determination Algorithm* to adjust the wear ranges dynamically.
- We propose an index structure to fast locate the appropriate *wear ranges*. The index is kept in DRAM to ensure high performance for allocation/deallocation.

The remainder of this paper is organized as follows. Section 2 discusses the architecture of in-memory file systems in a hybrid DRAM/NVM platform and the challenges in state-of-the-art in-memory file systems. Section 3 presents the design of *DWARM* scheme. Section 4 presents the experimental evaluation results. Section 5 presents related work. This paper is concluded in Section 6.

2. Background and motivation

The proposed memory management scheme targets hybrid computer systems that include non-volatile main memory (NVM) along with DRAM. In this section, we briefly describe the system architecture for existing in-memory file systems. We then discuss the challenges in the design for in-memory file systems.

2.1. System architecture

Non-volatile, byte-addressable memory can be directly attached to the memory bus, providing near-DRAM data access speed. Recent studies have proposed three alternative approaches on how NVM can fit into the memory hierarchy [4]. First, replacing DRAM with NVM to achieve large memory capacity. Second, using DRAM as another level of hardware cache for NVM. Third, combining DRAM and NVM in a unified address space.

Because of the limited endurance and high write latency, placing NVM alongside DRAM in a unified address space is an attractive approach to harness the advantages of NVM while mitigating its drawbacks [11]. Under this assumption, Fig. 1 shows the architectural overview of in-memory file systems.