Advanced Science and Technology Letters
Vol.51 (CES-CUBE 2014), pp.219-223
http://dx.doi.org/10.14257/astl.2014.51.50

Mapping Method for Hybrid Storage to Enhance Storage Utilization of SSD

In-Soo Bae[1],Yun-Su Lee[1], Seung-Kook Cheong[2], Yunsik Kwak[1], and Seokil Song[1]

[1]Department of Computer Engineering, Korea National University of Transportation, Chungju, Chungbuk,Korea
{gkdrmf23,yunsou.lee}@gmail.com, {yskwak, sisong}@ut.ac.kr
[2]Electronics and Telecommunications Research Institute, Daejeon, 305-700, Korea
skjeong@etri.re.kr

Abstract. In this paper, we propose a new mapping method for a hybrid storage which consists of a SSD and a few HDDs to achieve low cost and high performance storages. Some existing hybrid storages use extents to manage a large storage efficiently. However, they cause an extent fragmentation problem. The extent fragmentation reduces the SSD utilization and the overall performance of the hybrid storage. In this paper, to solve the problem we propose a new mapping method that uses different size of IO unit for SSD and HDDs. Also, we perform simulations to verify our proposed method.

Keywords: hybrid storage, SSD, mapping method.

1 Introduction

Even though SSDs are becoming cost effective it is rather early that they replace HDDs in enterprise environment. Subsequently, several hybrid storage systems [1, 2, 3, 4] that combine a small SSD with HDDs have been proposed to exploit the IO performance of SSDs to enhance the overall IO performance with comparatively low cost. Usually, a hybrid storage system consists of a small and fast solid state drive (SSD) and one or more slow and large hard disk drives (HDD).

HCMSS (Hot Cold Management Storage System) [5] is one of the recently proposed hybrid storage system. It organizes a DDR-SSDs [6] and HDDs on the same layer. Frequently accessed data are placed on SSDs and data blocks are migrated between SSDs and HDDs according to their access frequency and recency. The mapping manager of HCMSS manages storage space in extents. When HCMSS receives IO requests from VFS, and it maps the IO requests to a physical extents in a SSD or a HDD. Sometimes, it allocates new extents from a SSD or a HDD, or migrates extents in a device to the other device by considering their temperature. An extent size is a multiple of the page size and it is given when creating a hybrid block device. Extents may decrease the management cost (the size of main memory for

[1]50 Daehak-ro, Chungju-si, Chungbuk 380-702, Corresponding Author, sisong@ut.ac.kr.

ISSN: 2287-1233 ASTL

mapping and free space management) of HCMSS, and increase the IO performance of HDDs.

However, the extent mapping method may cause an extent fragmentation problem. In Figure 1, the extent fragmentation problem is described. In this figure, we assume that the block size is 4kbytes and the extent size is 64kbyte (16 blocks). As shown in this figure, only 4 blocks in extent 0 are used, i.e., only 4 blocks have valid data, and we call this extent as a fragmented extent. There are 4 fragmented extents such as extent 0, 1, 3 and 4 in this figure, and 40% of the SSD is not used even though the SSD is full. Consequently, a new write request from VFS cannot be processed in the SSD.

The extent fragmentation problem arises by random writes of small files, and it is becoming more serious when small files are created and deleted repeatedly. Since we assume that the size of HDD is very large, the fragmentation problem of HDDs may not impact the overall performance of HCMSS. However, the problem of SSDs which is relatively small is critical for the performance of HCMSS. The extent fragmentation reduces the SSD utilization and the overall performance of the HCMSS.

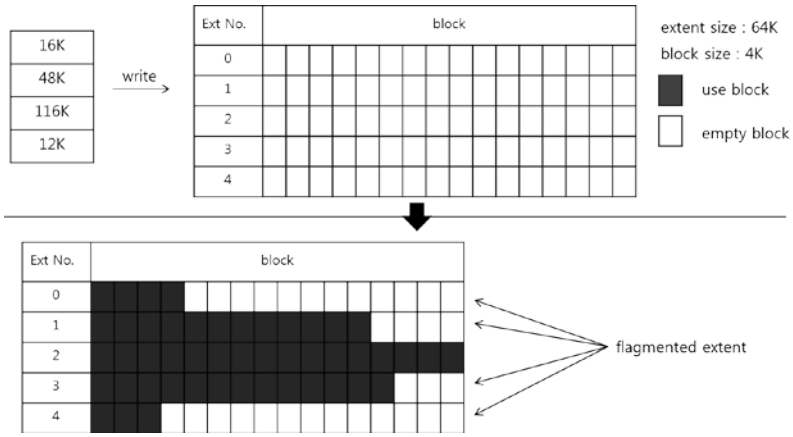

Fig. 1. Extent fragmentation problem

In this paper, we propose a new mapping table for the HCMSS to solve the extent fragmentation problem of SSD. We propose a new mapping method that uses different size of IO unit for SSD and HDDs, respectively. Also, we perform simulations to verify our proposed method. This paper is organized as follows. In Section 2, we describe the proposed mapping method and in Section 3, the performance results are presented. Finally, section 4 concludes this paper.

2Proposed mapping method

Figure 2 shows that the architecture of the proposed mapping method. Basically, the proposed mapping method is an enhanced version of HCMSS's mapping method. Unlike the HCMSS's mapping method, we add a block level mapping table and a free

space bitmap for SSD, and also we change the structure of the original mapping table by adding a bitmap field. The bitmap field of the mapping table is for indicating that each block of an extent is used or not. The HDD bitmap indicates that each extent is used or not. On the other hand, the SSD bitmap indicates that each block of SSD is used or not. While the mapping table maps block IO requests that contains logical block number to physical extents in HDDs, the SSD hash table maps block IO requests to physical blocks in SSD.

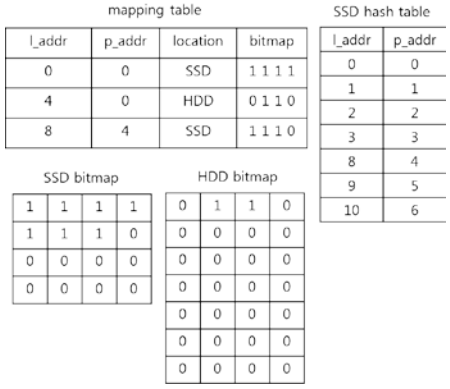

Fig. 2. Architecture of Proposed Mapping Method.

In Figure 2, the block size is 4kbytes and the extent size is 16kbytes, so the bitmap field of the mapping table has 4 bits for the corresponding 4 blocks of an extent. The SSD contains 16 blocks, and the HDD have 28 extents (112 blocks). The logical extents 0 and 1 are placed on the SSD, and the logical extent 5 is placed on the HDD. For the extents 0 and 1 on the SSD, we manage block level mapping information in the SSD hash table.

When a block request is given by VFS, our mapping manager checks that the block is on the SSD or the HDD by searching the mapping table. If the block is on the SSD, the mapping manager maps the block to a physical block on the SSD by searching the SSD hash table. Otherwise, the mapping manager maps the block to an extent on the HDD. In some cases, the mapping operation for the block IO request may fail. It means that the block IO request is a new write operation. In that case, the mapping manager tries to allocate a block from the SSD and if it fails, it allocate an extent from the HDD. When some extents of the HDD need to be migrated, the mapping manager moves only the blocks which corresponding bit is 1. On the contrary, when some blocks of the SSD need to be migrated, extents for the blocks are allocated from the HDD and the blocks are moved to those extents.

3Performance Evaluation

In this paper, we evaluate the performance of our proposed mapping method through simulation. We compare our proposed method with the mapping method of the

HCMSS in terms of the storage utilization. The simulation parameters used in our experiments are shown in Table 1. The size of IO requests and their logical block addresses are randomly generated. Also, the HCMSS consists of a SSD and a HDD.

Table 1. Simulation parameters.

parameters	Value
Block Size	4Kbyte
Extent Size	64Kbyte
I/O Size	4~256Kbyte
I/O Count	10000
Migration Area	30%(random)

Figure 3 shows the storage utilization of a SSD space utilization with varying the size of IO requests. As shown in the figure, the storage utilization of our proposed method is much better as when the IO size is becoming smaller. That is, the proposed mapping method works well even though IO requests are random writes for small files.

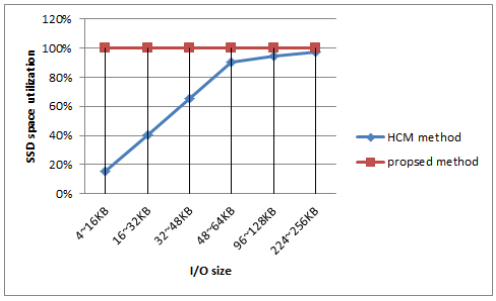

Fig. 3. Storage utilization of SSD with varying the size of IO requests.

4Conclusion

In this paper, we proposed a new mapping method for the HCMSS to solve the extent fragmentation problem. The proposed method used hybrid approach that manages SSD and HDD with different IO unit size. The proposed mapping method increased the storage utilization of the SSD up to 4 times when the IO size is 4 or 16kbytes comparing to the original mapping method of the HCMSS.

References

1. Badam, A. and Pai, V. S.: SSDAlloc: Hybrid SSD/RAM memory management mad easy. Proceedings of USENIX NSDI, pp. 211--224 (2011)

2. Wu, X. and Reddy, A. L.: Exploiting concurrencyto improve latency and throughput in a hybridstorage system. Proceedings of AnnualIEEE/ACM International Symposium on Modeling. pp. 14--23 (2010)
3. Koltsidas, I. andViglas, S. D.: Flashing up thestorage layer. Proceedings of VLDB Conference. pp. 514--525 (2005)
4. Jo,H. , Kwon,Y., Kim,H.,Seo,E., Lee, ,J. and Maeng, S.: SSD-HDD-Hybrid virtual disk inconsolidated environments. Proceedings ofInternational Conference on Parallel Processing.pp. 375--384 (2013)
5. Ki-jeongKhil, Seokil Song, Seung-Kook Cheong: Hot and Cold Data Replacement Method for Hybrid Storage System. An International Interdisciplinary Journal. (ISSN: 1343-4500) Vol. 16, No. 12(A), pp. 8331--8337 (2013)
6. J. Hwang and S.Chung: Technology trend of DDR based SSD storage system. NIPA Weekly Trend Report, Vol. 1421, pp. 28--41 (2009)
7. Yunsoo Lee, Dongmin Shin, InsooBae, Seokil Song, Seungkook Cheong: Time Stamp based Multiple Snapshot Management Method for Storage System. CES-CUBE 2013, ASTL Vol. 25, pp. 251—255 (2013)

