The Journal of The Institute of Internet, Broadcasting and Communication (IIBC)
Vol. 19, No. 4, pp.65-69, Aug. 31, 2019. pISSN 2289-0238, eISSN 2289-0246

https://doi.org/10.7236/JIIBC.2019.19.4.65
JIIBC 2019-4-10

응용의 특성을 고려한 NVM 기반 고속 스토리지의 배치 방안

Allocation Techniques for NVM-Based Fast Storage Considering Application Characteristics

김지선*, 반효경**

Jisun Kim*, Hyokyung Bahn**

요 약 본 논문은 응용의 특성을 고려하여 NVM 기반 고속 스토리지를 최적 배치하는 방안을 논의한다. 이를 위해 본 논문은 먼저 여러 응용들의 스토리지 접근 특성을 분석하여 효율적인 NVM 배치에 활용가능한 다음의 두 가지 특성을 관찰하였다. 그 첫째는 I/O를 집중적으로 발생시키는 스토리지 파티션이 하나로 고정되지 않고 응용에 따라 다르게 나타난다는 점이다. 두 번째는 스토리지 접근에 있어 높은 비율의 1회성 접근 데이터가 존재한다는 점이다. 이와 같은 분석 결과를 토대로 본 논문에서는 NVM을 특정 파티션으로 고정 사용하는 것이 아니라 응용의 특성에 맞게 배치하는 것이 스토리지 성능을 극대화할 수 있음을 확인하였다. 특히, 그래프, DB, 웹 응용의 경우 NVM을 스왑, 저널, 파일시스템 파티션으로 활용하는 것이 효과적임을 확인하였다.

Abstract This paper presents an optimized adoption of NVM for the storage system considering application characteristics. To do so, we first characterize the storage access patterns for different application types, and make two prominent observations that can be exploited in allocating NVM storage efficiently. The first observation is that a bulk of I/O does not happen on a single storage partition, but it is varied significantly for different application categories. Our second observation is that there exists a large proportion of single accessing in storage data. Based on these observations, we show that maximizing the storage performance with NVM is not obtained by fixing it as a specific storage partition but by allocating it adaptively for different applications. Specifically, for graph, database, and web applications, using NVM as a swap, a journal, and a file system partitions, respectively, performs well.

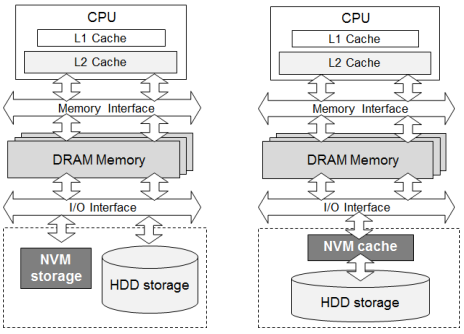
Key Words : Fast Storage, NVM, Application, File system, journal area, swap area

Ⅰ. 서 론

최근 상변화메모리, 마그네틱메모리 등 고속 스토리지로 활용 가능한 NVM 매체가 출현하면서 이들 매체가 스토리지 시스템의 구성에 활용될 가능성이 높아지고 있다[1,2]. NVM을 활용한 파일시스템, 저장장치 캐쉬 등 다양한 연구가 진행되고 있지만, 한정된 NVM을 파일시스템, 저널 영역, 스왑 영역 등 스토리지 구성 중 어떤 영역으로 활용하는 것이 시스템의 성능을 극대화할 수 있는지에 대해서는 아직까지 뚜렷한 답을 얻지 못하고 있다[3,4].

*비회원, 이화여자대학교 컴퓨터공학과
**정회원, 이화여자대학교 컴퓨터공학과(교신저자)
접수일자 2019년 5월 31일, 수정완료 2019년 7월 2일
게재확정일자 2019년 8월 2일

Received: 31 May, 2019 / Revised: 2 July, 2019 /
Accepted: 2 August, 2019
*Corresponding Author: bahn@ewha.ac.kr
Dept. of Computer Engineering, Ewha University, KOREA



(a) 하이브리드 스토리지 (b) 스토리지 캐쉬

그림 1. NVM을 이용한 스토리지 구성
Fig. 1. Storage architecture by adopting NVM.

본 논문은 이러한 질문에 답하기 위해 다양한 응용별 스토리지 접근 특성을 분석하고 이에 기반한 NVM 스토리지의 최적 활용 방안에 대해 논의한다. NVM을 스토리지에 활용할 수 있는 방안은 그림 1에서 보는 것처럼 기존의 하드디스크와 NVM을 하이브리드 스토리지 형태로 사용하는 방법과 NVM을 하드디스크의 캐쉬 형태로 사용하는 방법이 있다. 본 논문은 스토리지 접근에 있어 1회성 참조 데이터가 많은 양을 차지함을 분석하고 이에 기반하여 NVM을 하이브리드 스토리지로 활용하는 것이 효과적임을 보인다.

한편, 컴퓨터 시스템이 사용하는 스토리지 영역은 크게 파일 시스템, 저널 영역, 스왑 영역 등으로 나누어 볼 수 있다[5,6,7,8]. 본 논문에서는 다양한 응용의 스토리지 접근 특성을 분석한 결과 잦은 스토리지 접근을 발생시키는 영역이 파일시스템, 스왑 영역, 저널 영역 중 하나로 고정되지 않고 응용 별로 상이하다는 점을 확인하였다.

특히, DB 응용에서는 트랜잭션 단위의 관리가 필요하므로 저널 영역에 지속적인 쓰기 연산이 발생하여 저널 영역이 가장 많은 스토리지 접근을 초래하였다. 반면 그래프 응용의 경우 메모리 사용량이 크기 때문에 스왑 영역이 전체 스토리지 접근 중 가장 많은 부분을 차지하였다. 이는 메모리 용량이 응용의 작업 집합을 수용하기에 충분하지 않을 경우 잦은 스왑 영역 접근이 불가피하기 때문이다. 이에 비해 웹 응용에서는 데이터 파일 접근이 많기 때문에 파일 시스템 접근이 가장 많은 스토리지 I/O를 초래하였다.

이와 같은 분석 결과를 토대로 고속의 NVM이 주어졌을 때 파일시스템, 저널 디바이스, 스왑 디바이스로 사용시 각 응용의 스토리지 성능 개선 효과가 어느 정도인지를 실험을 통해 분석하였다. 실험 결과 NVM을 고정된 용도의 스토리지 파티션으로 사용할 경우 성능 개선 폭이 크지 않으며, 응용 특성에 맞게 할당하는 것이 필요함을 확인하였다. 그래프 응용의 경우 NVM을 스왑 디바이스로 활용할 때 가장 큰 성능 개선을 나타내었으며, DB 응용의 경우 저널 디바이스로 활용할 경우 가장 좋은 성능을 나타낼 것이라는 예상과는 달리 NVM의 용량이 매우 작을 경우 파일 시스템으로 할당 했을 때 더 좋은 성능을 나타내는 것을 확인할 수 있었다. 이는 DB 응용의 경우 저널링 입출력이 많은 스토리지 접근을 발생시키지만 접근 빈도가 매우 높은 인기 데이터가 파일 시스템에 존재하기 때문으로 확인되었다. 웹 응용에서는 NVM을 파일 시스템 용도로 활용했을 때 가장 좋은 성능을 내었다.

표 1 트레이스 분석에 활용된 스토리지 파티션 정보
Table 1. Storage partition information used in trace analysis.

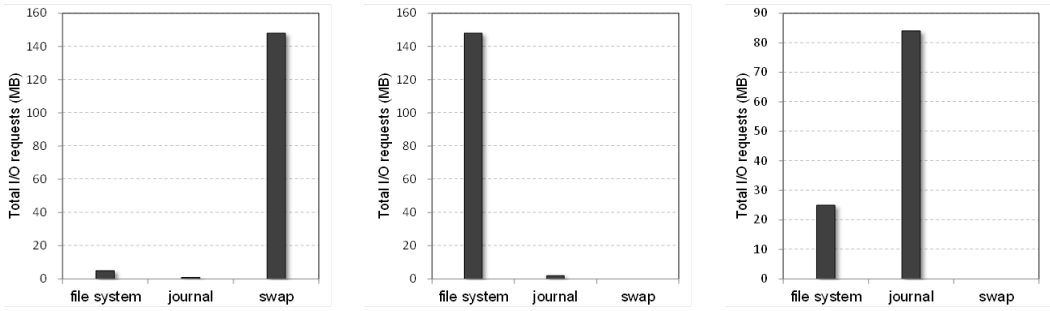
Partition	Start	End	Description
/dev/sda1	2048	103999487	File system
/dev/sda5	104001536	111998975	Journal device
/dev/sda6	112001024	119998463	Swap device

본 논문의 구성은 다음과 같다. II장에서는 응용에 따른 스토리지의 접근 패턴을 분석한다. III장에서는 NVM을 다양한 스토리지 계층으로 활용함에 따른 성능을 실험적으로 분석한다. 끝으로 IV장에서는 본 논문의 결론을 제시한다.

II. 응용의 스토리지 접근 분석

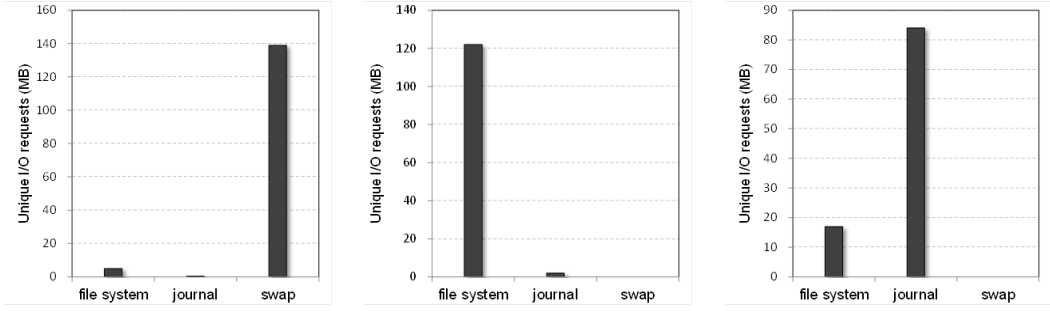
본 장에서는 응용들이 어떤 스토리지 영역에서 I/O를 주로 발생시키는지를 확인하기 위해 영역별로 스토리지 접근 트레이스를 추출하고 이를 분석한다. 각 영역의 구분을 위해 스토리지를 파일 시스템, 저널 영역, 스왑 영역으로 나누어 실험하였다. 실험 환경은 512MB DDR3-10700 메모리와 60GB SATA HDD로 구성하였고, Ubuntu 14.04 64 bit와 Ext4 파일시스템을 사용하였다. 저널 영역에서 발생하는 스토리지 접근을 별도로 추출하기 위해 Ext4의 외부 저널링(external journaling) 옵션을 사용하였다. 표 1은 본 논문의 실험에 사용된 스토리지 파티션별 정보를 보여주고 있다.

본 논문에서는 그래프 응용, DB 응용, 웹 응용으로 구성된 3종의 서로 다른 응용들에 대한 스토리지 접근 트레이스를 추출하였다. 그림 2는 각 파티션 별 스토리지 접근량을 보여 주고 있다. 그림에서 보는 것처럼 하나의



(a) 그래프 응용 (b) 웹 응용 (c) DB 응용

그림 2. 스토리지 영역별 접근량
Fig. 2. Total access traffic for each storage area.



(a) 그래프 응용 (b) 웹 응용 (c) DB 응용

그림 3. 스토리지 영역별 중복 배제된 접근량
Fig. 3. Unique access traffic for each storage area.

특정 파티션에서 스토리지 접근이 집중적으로 발생하지 않고, 응용에 따라 다르게 나타나는 것을 확인할 수 있다.

그림 2(a)에서 보는 것처럼 그래프 응용은 스왑 영역에서 가장 많은 스토리지 접근이 발생하였다. 그래프 응용은 그래프의 각 점들을 계산하고 이를 화면에 표시하기 위해 많은 양의 메모리를 필요로 하기 때문에 메모리 요구량이 응용의 작업집합을 수용하기에 충분하지 않을 경우 많은 스왑 영역 접근을 발생시키기 때문으로 분석된다. 그림 2(c)에서 보는 것처럼 DB 응용은 저널 영역에서 가장 많은 스토리지 접근을 발생시키고 있다. 이는 DB 응용이 트랜잭션 처리를 필요로 하므로 저널 영역에 많은 쓰기 연산을 발생시키기 때문이다. 한편 DB 응용은 파일시스템 영역에도 일정량 이상의 스토리지 접근이 발생하는 것을 확인할 수 있다. 이는 트랜잭션 처리 응용의 수행이 빈번한 저널링 I/O를 유발하지만 궁극적으로 파일 및 그 메타데이터에 대한 접근도 수반함을 의미한다. 그림 2(b)에서 보는 것처럼 웹 응용의 경우 파일 시스템 영역에서 가장 많은 I/O가 발생했다. 이는 웹 응용이 웹 서버로부터 파일을 읽어 이를 로컬 파일 시스템에 저장하고 필요시 다시 읽어 화면에 표시하는 등 파일 시스템 접근이 빈번하기 때문으로 파악된다.

그림 3은 그림 2와 동일한 트레이스를 사용하여 같은 데이터의 중복 접근을 배제한 스토리지 접근 비율을 보여주고 있다. 즉, 그림 3은 동일 데이터가 여러 차례 접근된 경우에도 이를 1번만 카운트한 결과를 나타낸다. 그림 2와 그림 3을 비교함으로써 각 파티션에 존재하는 데이터에 대한 상대적인 인기도를 파악할 수 있다. 예를 들어 그림 3에서의 그래프 높이가 그림 2보다 낮을 경우 해당 파티션은 인기 있는 데이터를 상대적으로 많이 보유하고 있음을 의미한다.

그림 2와 그림 3을 비교해 보면 그래프 응용, 웹 응용의 경우 전반적으로 비슷한 모양을 나타내어 특별히 인기있는 데이터를 보유한 파티션은 발견되지 않았다. 그러나, DB 응용의 경우 그림 3의 파일시스템 그래프가 그림 2에 비해 낮은 것을 확인할 수 있다. 이는 DB 응용의 경우 저널 영역에 많은 접근이 있으나 인기 있는 데이터는 파일 시스템 영역에 밀집되어 있음을 의미한다.

스토리지 접근에 대한 또 다른 중요한 분석 대상 중

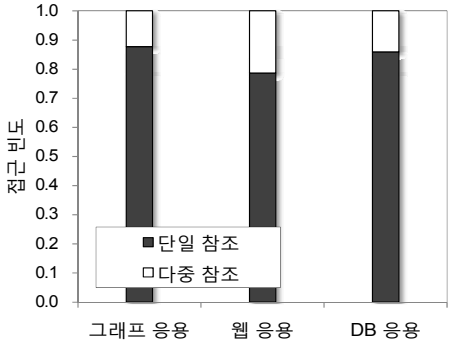


그림 4. 단일 참조와 다중 참조의 분포
Fig. 4. Distribution of single and multiple accesses.

하나는 데이터에 대한 재사용성 분석이다. 특히, NVM이 캐쉬로 사용될 경우 재사용되지 않는 데이터는 성능 개선 효과가 없으므로 데이터의 재사용성이 어느 정도인지를 분석하는 것은 중요하다. 이러한 특성을 분석하기 위해 본 논문에서는 응용별 스토리지 접근을 단일 참조와 다중 참조로 구분해서 그 비율을 확인해 보았다. 그림 4에서 보는 것처럼 단일 참조의 비율이 그래프 응용, 웹 응용, DB 응용 모두 80-90%에 이르는 것을 확인할 수 있다. 이러한 단일 참조 데이터는 NVM이 캐쉬로 사용될 경우 NVM에 보관하더라도 재사용이 되지 않으므로 성능 개선에 도움이 되지 않는다. 반면, NVM을 파일시스템, 저널, 스왑 등 말단 스토리지 영역으로 활용할 경우 단일 참조 데이터의 경우에도 느린 디스크 접근을 없애고 고속 NVM으로 스토리지 접근이 대체되므로 성능 개선 효과를 얻을 수 있다. 즉, NVM이 캐쉬로 사용될 경우 최초의 참조는 느린 스토리지 접근이 동반되나, NVM을 스토리지 자체로 사용할 경우 느린 스토리지 접근을 없애는 효과를 기대할 수 있다. 이러한 분석을 토대로 본 연구에서는 NVM을 캐쉬가 아닌 스토리지 파티션으로 활용하는 것이 더 성능 개선 효과를 높일 수 있다고 결론지을 수 있다. 다만, 스토리지 파티션 중 어떤 영역으로 사용하는 것이 성능 개선 효과를 극대화시킬 수 있는지는 응용에 따라 다를 것으로 보인다.

III. 실험 결과

본 장에서는 하드디스크와 NVM으로 구성된 하이브리드 스토리지와 하드디스크 기반 단일 스토리지의 성능을 응용별로 비교 분석한다. 그림 5는 하드디스크에 NVM이 추가됨에 따라 각 응용의 수행 시간이 어떻게 달

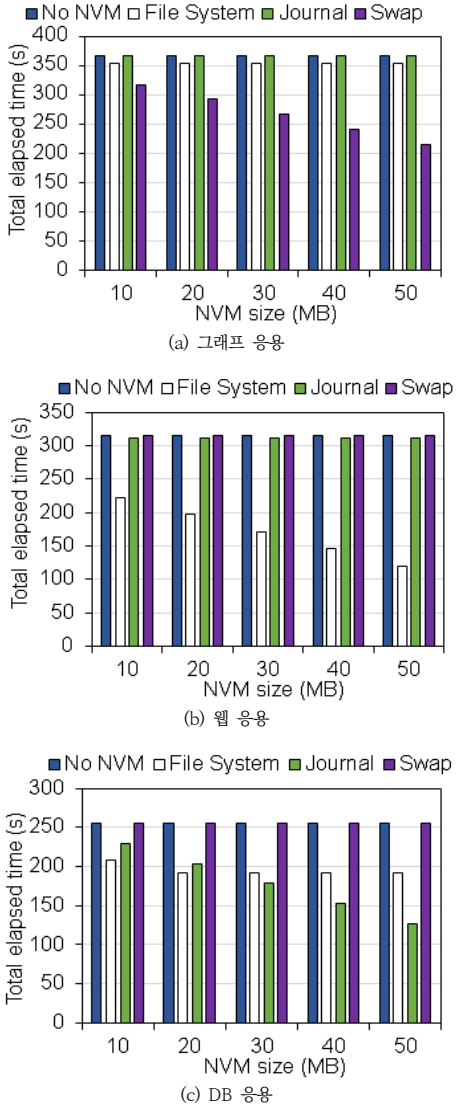


그림 5. 성능 평가 결과
Fig. 5. Performance evaluation results.

라지는지를 보여주고 있다. 추가된 NVM을 각각 파일시스템, 저널, 스왑 영역으로 사용했을 때의 성능과 하드디스크만 사용한 스토리지의 성능을 비교분석한 결과 응용에 따라 성능 개선 효과가 가장 큰 NVM의 구성이 상이한 것을 확인할 수 있다. 그림 5(a)에서 보는 것처럼 그래프 응용의 경우 NVM을 스왑 영역으로 사용할 때 가장 성능이 좋은 것을 확인할 수 있다. 이는 II장의 분석과도 일관성 있는 결과이다. 그림 5(b)에서 보는 것처럼 웹 응용의 경우 NVM을 파일 시스템으로 사용했을 때 성능이 가장 많이 개선되는 것을 확인할 수 있다. 끝으로 그림

5(c)에서 보는 것처럼 DB 응용의 경우 NVM 용량에 따라 최적의 결과를 나타내는 할당 방안이 다름을 확인할 수 있다. NVM 용량이 20MB 이하일 때는 파일시스템으로 사용했을 때 가장 좋은 성능을 나타내었으나, NVM 용량이 커짐에 따라 저널 영역으로 사용할 때 더 좋은 성능을 나타내었다. 이는 II장에서 논의된 것처럼 파일시스템 영역에 인기있는 소량의 데이터가 존재하여 소량의 NVM이 이를 흡수할 경우 가장 좋은 성능을 나타내나, NVM 용량이 커질 경우 대량의 I/O를 발생시키는 저널 영역으로 사용하는 것이 효과적임을 의미한다.

Ⅳ. 결 론

본 논문에서는 한정된 NVM이 주어질 경우 이를 어떤 용도로 사용하는 것이 스토리지의 성능 개선 효과가 가장 클 것인지에 대해 서로 다른 응용들의 I/O 패턴 분석을 통해 살펴보았다. 분석 결과 응용에 따라 많은 양의 I/O를 발생시키는 스토리지 영역이 다르게 나타나는 것을 확인하였다. 본 논문의 실험 결과를 통해 그래프 응용, 웹 응용, DB 응용의 경우 각각 NVM을 스왑 영역, 파일시스템 영역, 저널 영역으로 활용하는 것이 효과적임을 확인할 수 있었다. 다만, DB 응용의 경우 소량의 NVM을 파일시스템 내 핫데이터 흡수용으로 사용할 경우 추가적인 성능 개선이 가능함을 확인하였다.

References

[1] H. Wong, S. Raoux, S. Kim, J. Liang, J. Reifenberg, B. Rajendran, M. Asheghi, and K. Goodson, "Phase Change Memory," Proceedings of the IEEE, Vol. 98, No. 12, pp. 2201-2227, 2010.
DOI: https://doi.org/10.1109/JPROC.2010.2070050

[2] E. Lee, H. Bahn, and S. Noh, "Unioning of the Buffer Cache and Journaling Layers with Non-volatile Memory," 11th USENIX Conference on File and Storage Technologies (FAST13), pp. 73-80, 2013.

[3] R. Fang, H. Hsiao, B. He, C. Mohan, and Y. Wang, "High performance database logging using storage class memory," Proc. of IEEE ICDE, pp. 11-16, 2011.

[4] E. Lee, J. Jang, T. Kim, and H. Bahn, "On-demand Snapshot: An Efficient Versioning File System for Phase-Change Memory," IEEE Transactions on Knowledge and Data Engineering, Vol. 25, No. 12, pp.2841-2853, 2013.
DOI: https://doi.org/10.1109/TKDE.2013.35

[5] S. Lee, H. Bahn, S. Noh, "CLOCK-DWF: A Write-History-Aware Page Replacement Algorithm for Hybrid PCM and DRAM Memory Architectures," IEEE Transactions on Computers, vol. 63, no. 9, pp. 2187-2200, 2014.
DOI: https://doi.org/10.1109/TC.2013.98

[6] S. Nam, H. Bahn, "Real-time Task Scheduling Methods to Incorporate Low-power Techniques of Processors and Memory in IoT Environments," The Journal of The Institute of Internet, Broadcasting and Communication, Vol.17, No.2, pp.1-6, 2017.
DOI: https://doi.org/10.7236/JIIBC.2017.17.2.1

[7] S. Min, Kwang. Lee, B. Jin, "A Design of Authority Management Protocol for Secure Storage Access Control in Cloud Environment," Journal of the Korea Academia-Industrial cooperation Society, Vol.17, No.9 pp.12-20, 2016.
DOI: https://doi.org/10.5762/KAIS.2016.17.9.12

[8] J. Jeong, H. Kim, S. Cheong, "A Study on the Performance Evaluation Tool of Hybrid Storage System Combined SSD and HDD," The Journal of KIIT, Vol.14, No.5, pp.131 - 136, 2016.
DOI: https://doi.org/10.14801/jkiit.2016.14.5.131

저 자 소 개

김 지 선(준회원)

- 2011년 2월 : 한신대학교 컴퓨터공학과 학사
- 2014년 3월 ~ : 이화여자대학교 컴퓨터공학과 대학원생
- 주관심분야 : 운영체제, 스토리지 시스템

반 효 경(정회원)

- 1997년 2월 : 서울대학교 계산통계학과 학사
- 1999년 2월 : 서울대학교 전산과학과 석사
- 2002년 2월 : 서울대학교 컴퓨터공학부 박사.
- 2002년 9월 ~ : 이화여자대학교 컴퓨터공학과 교수.
- 주관심분야 : 운영체제, 스토리지시스템, 임베디드시스템

※ This research was funded by the ICT R&D program of MSIP/IITP (2019-0-00074, Developing system software technologies for emerging new memory that adaptively learn workload characteristics) and also by the Basic Science Research program through the NRF grant funded by Korea Government (MSIP) (No. 2016R1A2B4015750).