메모리 파일 시스템을 보완하는 이차 저장장치 I/O의 분석

안준욱°, 김현준, 한환수
성균관대학교 정보통신대학
{ajw1507, hjunkim, hhan}@skku.edu

Analyzing Supplement Secondary Storage I/O for Memory-Based File Systems

Joonwook Ahn°, Hyunjun Kim, Hwansoo Han
College of Information and Communication Engineering, Sungkyunkwan University

요 약

컴퓨터 시스템은 중앙처리장치, 휘발성인 메인 메모리, 비 휘발성인 2차 저장장치의 구조를 수년간 유지하고 있다. 하지만 최근 메인 메모리로 사용중인 DRAM이 집적 한계에 도달하고 새로운 소자가 개발됨에 따라 새로운 비 휘발성 메모리 소자를 메인 메모리로 하고자 하는 연구가 진행되고 있다. 이러한 연구의 일환으로 메모리 파일시스템에 대한 연구가 많이 있다. 본 논문에서는 메모리 파일 시스템을 사용할 때 메모리 부족으로 인한 2차 저장장치로의 스왑 과정에서 발생하는 I/O를 분석하여 불필요한 오버헤드를 확인 및 개선방향을 제시하였다.

1. 서 론

컴퓨터 소자는 최근 수십 년간 급격한 발전을 거듭해 왔고, 현재의 중앙처리장치, 캐시, 메인 메모리, 2차 저장장치의 구조를 수년간 유지하고 있다. 현재 메인 메모리로 사용중인 DRAM은 매우 빠른 읽기/쓰기 성능을 보이지만, 휘발성의 특성으로 인해 데이터를 보존 할 수 없어서 HDD나 SSD와 같은 2차 저장장치를 필요로 한다. 하지만 DRAM의 개발이 집적 한계에 도달하였고, 이에 따른 대체 소자의 개발이 활발히 진행되고 있다. 지금까지 개발된 비 휘발성 메모리 소자는 DRAM에 비해 성능이 안 좋았으나 새롭게 개발되는 STT-MRAM과 같은 소자는 DRAM의 성능에 근접할 것으로 예상된다[1].

새롭게 개발된 비 휘발성 메모리를 컴퓨터에 적용하고자 하는 연구가 진행 중에 있으며[2], 이에 따른 컴퓨터 시스템에 많은 변화가 있을 것으로 예상된다. 그 중 특히 메모리의 비 휘발성 특징을 이용한 메모리 파일 시스템에 대한 연구가 활발히 진행 중이다[11].

기존의 블록 파일 시스템은 메인 메모리에 비해 느린 2차 저장장치의 성능으로 인해 성능 저하가 발생하며, 이를 극복하기 위해 지연쓰기와 같은 기능을 제공한다. 또한 메모리와 2차 저장장치 사이에서 파일의 신뢰성을 보장하기 위한 기법을 적용하여 성능을 저하시킨다. 반면, 메모리 파일 시스템은 2차 저장장치 I/O로 인한 성능저하가 없고, 2차 저장장치와 메모리 사이의 신뢰성을 고려 하지 않아도 되어 성능이 매우 뛰어나다. 하지만 메모리가 파일 시스템의 자료를 보관할 만큼 충분한 용량을 지원하기는 어려우며, 이를 위해 2차 저장장치로의 확장이 필요로 하다. 이는 2차 저장장치에 데이터의 원본을 두고 동기화를 필요로 하는 기존의 파일 시스템과는 다르며, 운영체제의 스왑과 같은 기법을 통해 2차 저장장치로의 확장이 가능하다.

하지만 이로 인해 발생하는 2차 저장장치 I/O는 파일 시스템의 성능 저하를 발생 시킨다. 따라서, 본 논문에서는 메모리 파일 시스템에서 스왑이 발생할 때 생기는 성능 저하 요인을 확인하고 이를 개선할 방법을 찾고자 하였다. 이를 위해 마이크로 벤치마크인 IOzone[9]을 동작하여 메모리 파일 시스템인 tmpfs[8]에서 스왑이 발생할 때의 2차 저장장치 I/O와 블록 파일 시스템인 ext4에서의 2차 저장장치 I/O를 비교하여 tmpfs에서 발생하는 오버헤드를 확인하였다.

2. 메모리 파일 시스템

메모리 파일 시스템은 파일을 2차 저장장치의 블록이 아닌 파일 캐시로 메모리에 저장하는 파일 시스템이다. 2차 저장장치 I/O로 인해 발생하는 성능저하가 없어 매우 빠른 성능을 보이지만 메모리의 휘발성으로 인해

본 논문은 한국 연구재단의 지원 (NO. 2012011602)과 지식경제부의 지원으로 산업원천기술개발사업 (10041244, 스마트TV 2.0 소프트웨어 플랫폼)으로 수행된 연구결과임

표 1 실험 환경

CPU	Intel Core i5-2400 3.10GHz
OS	Ubuntu kernel 3.11.4
Memory	4GB DRAM
Secondary Storage	256GB SSD

로그와 같은 임시파일을 저장하는 용도로 사용되거나, 메모리에 배터리를 추가한 형태의 시스템에서 특수한 목적으로 사용되어 왔다. 최근 비 휘발성 메모리에 대한 연구가 활발히 진행 됨에 따라 메모리 파일 시스템이 미래의 파일 시스템으로 주목 받고 있으며, 이에 따른 MRAMFS, BPFS, SCMFS, PMFS와 같은 새로운 파일 시스템이 개발되고 있다[3,4,5,6].

하지만 메모리의 용량은 2차 저장장치에 비해 작아서 파일 시스템이 사용하기에는 부족하다. 메모리는 지난 30년동안 바이트당 가격이 150만배 떨어졌으나, HDD는 같은 기간 동안 6000만배 떨어졌다[7]. 현재 메모리의 용량 대비 가격은 HDD에 비해 100배 이상 비싸며, HDD 및 다른 2차 저장장치의 가격 저하 속도는 메모리에 비해 더 빠르다.

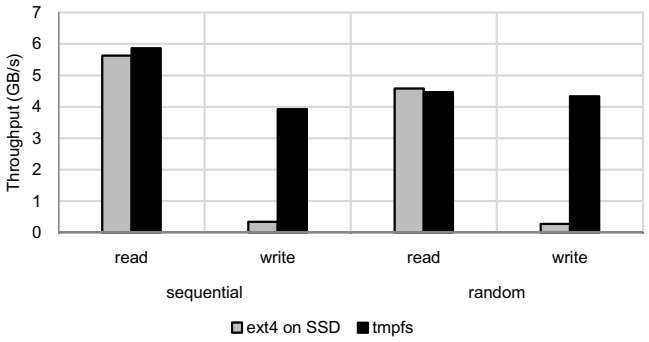
따라서 메모리 파일 시스템은 부족한 메모리 용량을 지원하기 위한 2차 저장장치로의 확장이 필요하다. 기존의 운영체제는 가상메모리상의 부족한 물리메모리를 지원하기 위해 2차 저장장치로의 스왑을 사용해 왔으며, 이를 메모리 파일 시스템에 적용할 수 있다.

본 논문에서는 메모리 파일 시스템으로 리눅스의 임시 파일 시스템인 tmpfs를 사용하였다. tmpfs는 공유 메모리영역을 사용하여 페이지 캐시에 파일을 보관하며, 운영체제의 스왑을 통해 2차 저장장치로의 확장을 지원한다. tmpfs는 스왑이 발생하지 않을 때 메모리 성능에 의해 매우 좋은 성능을 보이지만, 스왑이 발생하면 성능이 급격히 떨어진다. 이는 다음 장의 실험을 통해 확인해 볼 것이다.

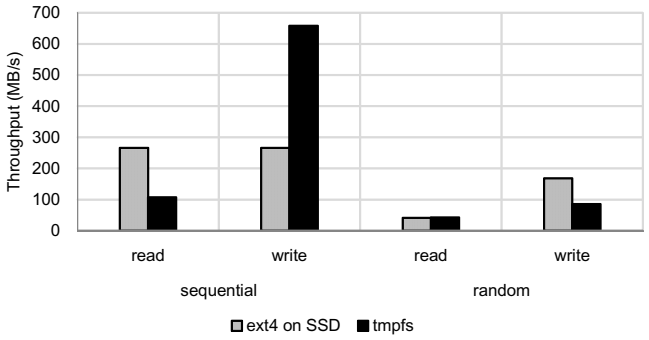
2.1. 메모리 파일 시스템의 성능

실험을 위해 표 1과 같은 환경에서 마이크로 벤치마크인 IOzone의 연속적 읽기/쓰기, 랜덤 읽기/쓰기 동작을 수행하여 성능을 측정하였다. tmpfs의 성능을 비교하기 위해 ext4를 사용하였으며 현재 tmpfs는 신뢰성 보장을 위한 모듈이 없기 때문에 ext4는 저널링이 없는 조건으로 실험을 진행 하였다.

그래프 1은 1GB의 데이터로 실험한 결과이다. tmpfs는 1GB의 실험에서는 스왑이 발생하지 않아서



그래프 1. IOzone 1GB data set 성능 측정



그래프 2. IOzone 5GB data set 성능 측정

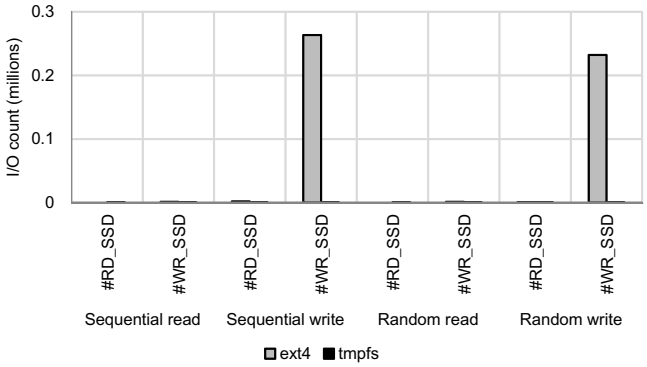
메모리 성능에 의해 매우 빠른 성능을 보인다. tmpfs는 쓰기 동작 수행 시 ext4에 비해 최대 15배 빠른 성능을 보이며, 읽기 동작은 동일한 페이지 캐시에서 동작을 수행하기 때문에 성능 차이가 없다.

그래프 2는 5GB의 데이터로 실험한 결과로, 메모리 용량보다 큰 실험조건으로 인해 tmpfs에서 스왑이 발생한다. tmpfs는 연속적 쓰기 동작에서 초기 비어 있는 메모리 공간으로 인해 성능이 좋게 나오는 것을 제외하고 ext4에 비해 성능이 안 좋게 나온다. 이는 tmpfs의 스왑 과정에서 불필요한 SSD I/O가 발생하여 성능을 저하시키기 때문이다.

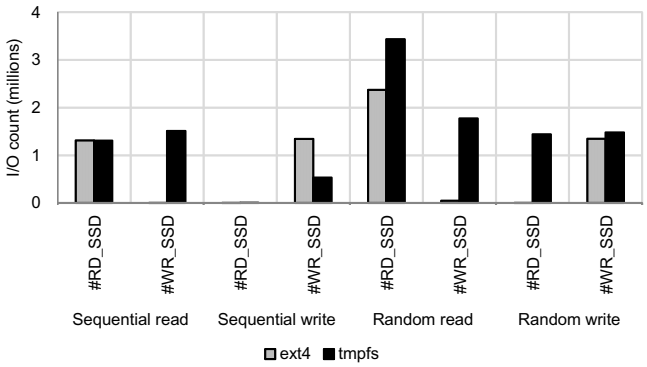
3. I/O 분석을 통한 성능 저하 요인 분석

앞선 실험에서 스왑이 발생 할 때 tmpfs의 성능이 급격히 저하되는 것을 확인 하였으며, 이를 확인 하기 위해 실제 커널에서 블록 I/O가 발생하는 부분을 수정하여 벤치마크가 동작하는 동안 발생하는 I/O를 기록하였다.

그래프 3은 IOzone 1GB 데이터 실험 결과로, ext4의 write에서 발생하는 SSD 쓰기를 제외하고는 SSD I/O가 거의 발생하지 않는다. 이는 그래프 1의 실험에서 ext4의 쓰기 동작에서만 tmpfs에 비해 성능이 안 좋은 것을 증명해 준다. ext4는 블록 파일 시스템으로 파일에 쓰여진 내용은 2차 저장장치에 반영이 되어야 하므로 SSD 쓰기가 반드시 발생한다. tmpfs는 페이지 캐시



그래프 3. IOzone 1GB data set SSD I/O



그래프 4. IOzone 5GB data set SSD I/O

내에서만 읽기/쓰기 동작을 수행하여 SSD I/O가 발생하지 않는 것을 확인 할 수 있다.

그래프 4는 5GB 데이터로 실험한 결과로, 스왑이 발생하여 tmpfs에서 SSD I/O가 발생하는 것을 확인 할 수 있다. ext4에 비해 tmpfs의 SSD I/O가 많이 발생하는 동작들이 있으며 추가적으로 발생하는 SSD I/O에 의해 tmpfs의 성능이 저하된다.

ext4에 비해 tmpfs에서 추가적으로 발생하는 I/O는 3가지로 분류할 수 있다. 1) 읽기 동작 수행 시 SSD 쓰기 발생, 2) 랜덤 쓰기 동작 수행 시 SSD 읽기 발생, 3) 랜덤 읽기 동작 수행 시 추가적인 SSD 읽기 발생.

읽기 동작에서 발생하는 SSD 쓰기의 경우 관련 연구[10]에서 제안한 tmpfs 페이지를 수정(modified) 또는 비수정(clean)으로 구분하여 해결 할 수 있을 것으로 예상된다. 랜덤 쓰기 동작에서 발생하는 SSD 읽기의 경우 스왑된 페이지를 쓰기 동작을 위해 메모리로 읽는 과정에서 발생하며, 연속적 쓰기동작은 파일을 생성하는 과정이기 때문에 스왑되어 있는 페이지가 없어서 SSD 읽기가 발생하지 않는다. 이는 쓰기동작을 수행 할 때 새로운 페이지를 할당함으로써 불필요한 SSD 읽기를 제거 할 수 있을 것이다. 또한 랜덤 읽기 동작에서 ext4에 비해 많이 발생하는 SSD 읽기의 경우, 스왑되어 있는 여러 개의 페이지를 한번에 메모리로 읽는 스왑 동작에 의해 발생하게 된다. 이는 메모리 파일 시스템의 페이지를 위한 새로운 스왑 메커니즘을 적용하여 해결할 수 있을 것 이다.

4. 결론 및 향후 연구

본 논문에서는 메모리 파일 시스템에서 스왑이 발생할 때의 성능저하와 이때 발생하는 2차 저장장치 I/O를 마이크로 벤치마크인 IOzone을 사용하여 확인하였다. 이를 바탕으로 IOzone의 각 동작에서 발생하는 성능 저하 요인을 확인 하였고 개선 방향을 제시 하였다. 향후 연구에서는 본 논문에서 찾은 불필요한 2차 저장장치(SSD) I/O를 제거하여 스왑이 발생할 때의 메모리 파일 시스템의 성능을 블록 파일 시스템에 대등하도록 구현할 예정이다.

5. 참고 문헌

[1] Driskill-Smith. Latest Advances and Future Prospects of STT-RAM. In Proceedings of Non-Volatile Memories Workshop(NVMW' 10). 2010.

[2] K. Bailey, L. Ceze, S. D. Gribble, and H. M. Levy. Operating system implications of fast, cheap, non-volatile memory, HotOS-13, 2011.

[3] Jeremy Condit, Edmund B. Nightingale, Christopher Frost, Engin Ipek, Benjamin C. Lee, Doug Burger, and Derrick Coetzee. Better I/O Through Byte-Addressable, Persistent Memory. SOSP, 2009.

[4] X. Wu and A. L. N. Reddy, SCMFS: A file system for storage class memory, SC, 2011.

[5] N. K. Edel, D. Tuteja, E. L. Miller, and S. A. Brandt, MRAMFS: A compressing file system for non-volatile RAM, MASCOTS, 2004.

[6] PMFS: Persistent memory file system, [online] https://github.com/linux-pmfs/pmfs/.

[7] J. C. McCallum. Market price survey. [online] http://www.jcmit.com/.

[8] Peter Snyder, tmpfs: A Virtual Memory File System. Proceedings of the autumn 1990 EUUG Conference, pp. 241-248, October 1990.

[9] W. D. Norcott. IOzone filesystem benchmark. [online] http://www.iozone.org/.

[10] Joonwook Ahn, Jungsik Choi, Hwansoo Han, Swap behavior for NVRAM file system, KCC 2013.

[11] Hyunjun Kim, Joonwook Ahn, Sungtae Ryu, Jungsik Choi, Hwansoo Han, In-Memory File System for Non-Volatile Memory, RACS 2013