차세대 저장 장치를 위한 메모리 확장

Memory Extension with Next-Generation Storage Device

한 혁
동덕여자대학교

Hyuck Han
Dongduk Women's University

요약

현대 운영 체제에서 가상 메모리 관리 기술은 응용 프로그램에게 가상의 큰 주소 공간을 제공하는 방법이다. 이러한 기술은 메인 메모리와 저장 장치를 이용하여 자주 접근되는 데이터는 메인 메모리에 덜 접근되는 데이터는 저장 장치에 저장한다. 본 연구는 차세대 저장 장치를 메모리 확장을 위한 장치로 가정했을 때의 성능 향상 기법을 제안한다. 본 연구진은 제안된 방법을 Linux 3.14.3을 구현하였고, 초고속 저장 장치를 이용하여 평가한 결과 기존 SWAP 시스템에 비해 20% 정도의 성능 향상 효과가 있음을 보였다.

I. 서론

최근 스마트폰, 태블릿과 같은 모바일 휴대 기기의 급격한 증가로 인해 기존 인터넷 서비스 시스템들이 효율적인 서비스 요청 처리를 위해 대용량 메모리를 사용하는 소프트웨어를 사용하기 시작했다. 예를 들어 OLTP 분야에서는 인 메모리 데이터베이스, 클라우드 컴퓨팅 분야에서는 memcached [1] 및 mongoDB와 같은 NoSQL 서버들이 사용되고 있다. 2008년 Facebook의 경우 빠른 데이터 요청 처리를 위해 800여대의 서버를 이용하여 28TB 정도의 memcached 기반 메모리 캐쉬를 사용해서 폭발적인 데이터 요청에 대응했다. 이 밖에도 가상 데스크탑 인프라(VDI) 분야 및 HPC 분야에서도 요구하는 메모리가 폭발적으로 증가하는 등 대용량 메모리를 요구하는 분야가 크게 늘어가고 있다.

그러나 현재 DRAM은 공정 미세화의 물리적 한계에 봉착하였고, 이에 따라 DRAM으로 메모리 요구량 증가에 대응하는 것은 가격적인 측면을 고려할 때 현실적으로 어려운 실정이다. 이러한 상황에 대응하기 위해 최근 몇 년 사이에 Flash SSD 기반 저장 장치를 활용하여 시스템 메모리를 확장하는 SWAP과 MMAP 기반 기술들이 개발되었다[2,3,4].

하지만 메모리를 확장시키는 운영체제 컴포넌트인 SWAP을 최신 PCIe Flash SSD인 삼성전자의 XS1715에 최적화하여 성능을 평가한 결과 높은 접근 지연 때문에 낮은 성능을 보이고, Flash SSD를 SWAP을 위한 페이징 장치로 사용할 때 수명이 18개월 정도로 줄어드는 것으로 나타났다. 따라서 Flash 기반 SSD는 접근 지연 시간이 크다는 하드웨어적인 한계로 인해 접근 지역성(spatial locality)이 아주 큰 응용 외에는 페이징 장치로 사용할 수 없어 현재의 증가하는 메모리 요구량에 대응하기 어렵다.

최근 활발하게 연구/개발이 진행되고 있는 MRAM, PRAM과 같은 차세대 비휘발성 메모리는 DRAM에 비해 직접도가 최소 3배 이상 높고, 접근 속도는 궁극적으로 DRAM과 비슷해질 것으로 전망된다. 이러한 차세대 메모리 소자는 집적도와 성능에서 보이는 장점 뿐 만 아니라 비휘발성이라는 물성 측면에서의 장점도 가지므로, 저장 장치를 위해 사용될 수 있다. 본 논문은 이러한 차세대 비휘발성 메모리 기반 SSD와 같이 아주 짧은 접근 지연 시간을 가지는 차세대 저장 장치를 가정하여 메모리를 확장하는 기법을 제안하고자 한다.

Ⅱ. 설계 및 구현

1. SWAP 슬롯 할당 최적화

기존의 SWAP 슬롯을 할당하는 알고리즘은 선형 찾기 기반이다. SWAP 공간의 크기가 작고 SWAP을 위한 저장 장치가 HDD와 같이 느린 장치라면 시스템에서 SWAP 슬롯을 할당하는 계산 비용은 크지 않다. 그러나 SWAP 공간의 크기가 크고 저장 장치의 접근 지연 시간이 매우 작다면 시스템에서 SWAP 슬롯을 할당하는 비용은 큰 오버헤드가 된다. 예를 들어 SWAP 공간의 크기가 32GB라면 SWAP 슬롯은 약 800만개 정도 생성되고, 응용이 큰 메모리를 지속적으로 사용한다면 800만개의 슬롯에서 선형적으로 할당받는 것은 문제가 된다. 실제로 본 연구진은 SWAP 슬롯을 할당하는데 CPU의 50% 정도의 사이클을 사용했음을 관찰하였다.

따라서 이러한 문제를 해결하기 위해 본 연구진은 SWAP 슬롯을 선형으로 관리하는 것이 아니라 tree 기반

자료 구조로 관리하는 것을 제안한다. Tree 기반 자료 구조로 관리하여 O(n)의 비용이 걸리는 슬롯 할당을 O(logn)의 비용으로 슬롯을 할당할 수 있다. 구현을 위해 SWAP 슬롯을 위한 tree의 차수를 8로 하였고 tree의 노드는 1B에 코딩하였다. 즉 tree의 말단 노드의 1bit가 하나의 SWAP 슬롯을 의미한다. 이런 방법을 사용하면 32GiB의 SWAP 공간을 위해 약 2MiB 정도의 메모리 공간을 차지하는 tree가 만들어진다.

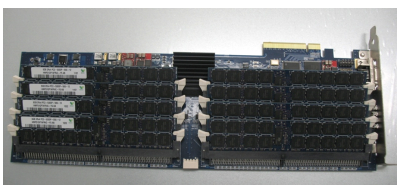
2. 그 밖의 최적화

본 연구진은 SWAP 슬롯을 할당하는 알고리즘의 최적화 이외에도 SWAP의 성능을 위해 read-ahead, I/O 스케줄러, Swappiness 등을 최적화하였다. Read-ahead는 과도한 I/O를 유발할 수 있기 때문에 read-ahead 기능은 사용하지 않는다. I/O 스케줄러는 차세대 저장 장치의 경우에는 스케줄링에 필요한 계산이 I/O 비용보다 커지게 되어서 I/O 스케줄러는 사용하지 않는다. Swappiness는 파일에 매핑된 페이지와 아무 파일에도 매핑되지 않는 페이지(SWAP)들 중에서 어느 페이지들을 어느 정도 많이 회수해야 하는지를 조정하는 파라미터이다. SWAP 시스템을 최적화하기 위해 swappiness 파라미터를 0으로 했고, 이는 가능한 파일에 매핑된 페이지들부터 회수하는 것을 의미한다.

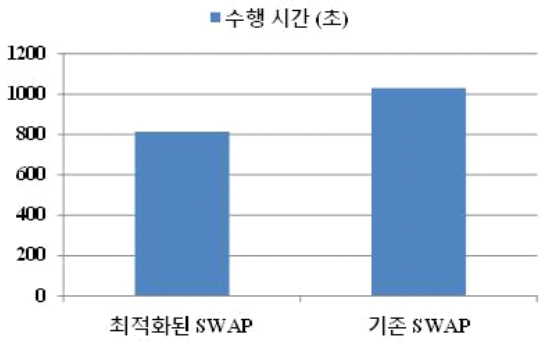
Ⅲ. 실험 및 평가

실험을 위해 Intel Xeon CPU E5630 2.53GHz를 장착한 서버 시스템을 사용하였다. 이 시스템은 8개의 코어와 24GiB 메인 메모리를 가지고 있으며, Linux 커널은 3.14.3을 사용하였다. 초고속 저장 장치를 위해 DRAM을 미디어로 사용하는 그림 1과 같은 DRAM SSD를 사용하였다[5]. 실험을 위해 과학 계산 벤치마크인 NPB를 사용하였고, 본 논문에서는 NPB의 ft 응용을 수행하였다. ft는 Fast Fourier Transformation (FFT)를 x, y, z 축으로 연속하여 수행하는 응용이다. ft 응용이 약 1.4GB의 메모리를 사용하기 때문에 본 연구진은 SWAP을 사용하는 환경을 만들기 위해 메인 메모리를 1GB만 사용하게 시스템을 설정하였다.

그림2는 ft 응용의 수행 시간을 나타낸 결과이다. 본 연구에서 개발한 최적화된 SWAP은 ft를 수행하는데 813초 그리고 기존 SWAP은 1029초 정도가 걸렸고, 최적화된 시스템이 21% 정도 성능 향상 효과가 있었다.



▶▶ 그림 1. 실험에 사용된 DRAM 기반 SSD



▶▶ 그림 2. ft 수행 시간 비교

Ⅳ. 결론

본 연구는 PRAM/MRAM와 같은 차세대 비휘발성 메모리 기반 SSD와 같이 초고속 저장 장치와 SWAP을 최적화하여 시스템 메모리를 확장하는 기술을 제안하였다. 본 연구진은 SWAP 슬롯 할당 알고리즘을 효율적으로 재설계 및 구현하였고 SWAP 관련된 시스템 파라미터를 최적화하였다. 이러한 최적화된 SWAP 시스템은 차세대 저장 장치 환경에서 향상된 성능을 보여주었다.

▌참 고 문 헌▐

[1] Intel, “Enhancing the scalability of memcached,” August 2012.

[2] M. Saxena and M. M. Swift, “Flashvm: Virtual memory management on flash.” USENIX Annual Technical Conference, ATC, June 2010.

[3] A. Badam and V. S. Pai, “Ssdalloc: hybrid ssd/ram memory management made easy.” USENIX conference on Networked systems design and implementation, NSDI, March 2011.

[4] B. Van Essen, H. Hsieh, S. Ames, and M. Gokhale, “Di-mmap: A high performance memory-map runtime for data-intensive applications,” in High Performance Computing, Networking, Storage and Analysis (SCC), 2012 SC Companion.

[5] TailwindStorage, “Extreme 3804, http://tailwindstorage.com/products/.”