SCM을 위한 제자리 쓰기 기반의 스냅샷 지원 기법

이은지○ 유승훈 장지은 고 건
서울대학교 컴퓨터공학부
{ejlee, shyoo, jejang, kernkoh}@oslab.snu.ac.kr

Designing a Write In-place Snapshot File System for Storage Class Memory

Eunji Lee○ Seunghoon Yoo Jee-eun Jang Kern Koh
School of Computer Science and Engineering, Seoul National University

1. 서 론

스토리지 클래스 메모리(SCM, Storage Class Memory)는 FeRAM, MRAM, PCM과 같이 비휘발성이면서 바이트 단위 접근이 가능한 메모리 기술을 총칭한다. 현재까지는 SCM의 높은 가격으로 인해 소량의 SCM을 성능 및 신뢰성 향상을 위한 용도로 사용하는 방법이 주로 연구되었으나 최근 SCM 집적도 기술의 급속한 발전과 함께 플래시 메모리를 대체하는 스토리지 레벨에서 SCM을 활용하는 방안이 연구되고 있다[1]. 최근 연구에 따르면 대용량화가 가시화되고 있는 PCM 등이 전력 및 공간 문제로 현재 한계점에 놓여있는 DRAM과 하드디스크의 대체 저장매체로 사용될 것으로 전망되고 있다[2].

이에 본 논문에서는 SCM이 스토리지로 사용하는 환경에서 효율적으로 동작하는 스냅샷 파일시스템을 제안한다. 스냅샷 파일시스템은 파일시스템의 일관성을 보장하고 과거 버전에 대한 접근을 지원하는 장점이 있어 현재 서버환경에서 널리 사용되고 있다. 대표적으로 WAFL, btrFS, ZFS 등이 이에 해당된다. 그러나 이러한 스냅샷 파일시스템들은 디스크 기반 환경에 맞게 설계되어 있어, 하드디스크와는 상이한 특성을 지니는 SCM의 장점을 제대로 활용하지 못하는 단점이 있다. 구체적으로, 현재의 스냅샷 파일시스템은 데이터가 업데이트 될 때 새로운 데이터를 원래 데이터가 있던 곳에 쓰지 않고 새로운 위치에 쓰는 "Copy-on-write(쓰기 시 복사)" 정책을 사용한다. 이것은 하드디스크의 대용량성을 이용해 헤드 이동에 따르는 탐색시간을 최소화함으로써 성능을 향상시키도록 설계된 것이다. 그러나 SCM은 탐색시간이 없고 동일한 접근 시간을 제공하기 때문에 이러한 Copy-on-write의 효력을 상실한다. 오히려 Copy-on-write의 연쇄적 포인터 업데이트로 인한 공간 사용량의 증가는 상대적으로 비싼 SCM에서 비용을 증가시키는 주요원인이 될 수 있다. 본 논문에서는 SCM의 동일한 속도의 랜덤접근이 가능하다는 특징을 활용하여 공간 효율성을 높이는 스냅샷 파일시스템을 설계하였다. 공간 사용량 증가의 주된 원인이 "다른 자리 쓰기(Out-place-update)"에 의한 연쇄적 포인터 업데이트임을 감안하여, 본 논문에서는 "제자리 쓰기(In-place-update)"에 기반한 스냅샷 형성 매커니즘인 WISP(Write In-place Snapshot Policy)을 제안한다.

2. 본 론

WISP은 새로운 데이터를 원래의 위치에 쓰고, 대신 원래의 데이터를 새로운 위치에 옮겨 둔다. 따라서 업데이트 이후에도 데이터가 동일한 물리적 위치에 존재하기 때문에 연쇄적 업데이트로 인한 공간 소비를 절약할 수 있다. 그림 1은 본 논문에서 제안한 WISP과 Copy-on-write방식에 기반한 스냅샷 파일시스템을 공간 효율성 측면에서 비교한 것이다. 트리구조의 파일시스템에서 리프노드 D가 업데이트 될 때, Copy-on-write는 데이터 블록 D, inode블록, root블록의 세 노드에 대한 공간이 필요한 반면, 본 논문에서 제안된 WISP은 데이터 블록D에 필요한 한 노드 공간만을 사용한다. 이는 Copy-on-write 공간 소비의 1/3에 불과하다. 한편 제안된 WISP 파일시스템은 과거 스냅샷 버전에 대한 접근을 지원하기

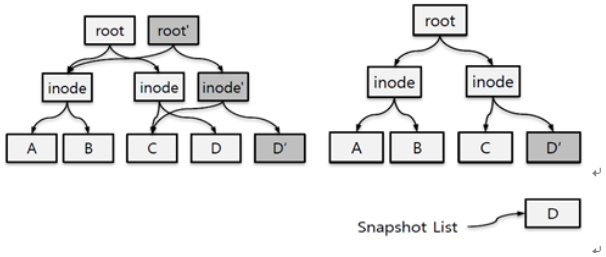

그림 1. Copy-on-write와 WISP의 공간사용량 비교

위해서는 추가적인 트리 복구작업을 수행해야 한다. 그림 1에서 보여주듯이 기존의 Copy-on-write는 이전 버전의 파일시스템에 대한 루트를 찾기만 하면 과거 스냅샷에 대한 접근이 가능했다. 그러나 WISP은 스택 형태로 저장된 데이터의 히스토리를 이용해 요청된 버전의 파일시스템을 일시적으로 생성하는 작업이 필요하다. 제안된 복구 매커니즘은 "Reverse copy-on-write(역 방향 쓰기 시 복사)" 인데, 이것은 마치 현재의 파일시스템 데이터가 과거 버전으로 업데이트되는 것처럼 여기며 Copy-on-write를 적용하는 것이다. 이러한 방식은 루트 블록 마운트를 통해 스냅샷 접근을 허용하는 Copy-on-write 정책에 비해 스냅샷 접근 시간을 지연시킬 수 있으나, 시스템 복구는 갑작스러운 전력소실과 같은 예외적인 상황에서만 발생하기 때문에 공간 효율성을 위한 스냅샷 접근 성능의 저하 간의 트레이드 오프는 합리적인 것이라 할 수 있다. 이 장에서는 WISP(Write In-place Snapshot Policy)에서 기본연산들의 구체적인 작업 절차와 요청이 발생하였을 때의 스냅샷 접근 방법에 대하여 설명한다.

트레이스 기반 시뮬레이션을 통해 Copy-on-write 방식과 제안된 스냅샷 기법의 공간사용량을 비교하였다. Varmail, File server, Web server, Video server, Proxy server 등의 다섯 가지 워크로드를 추출하였으며 최대 스냅샷 지원 수는 btrFS와 동일하게 256개로 가정하였다. 스냅샷 버전의 최대 수를 유지할 때 스냅샷에 의해 점유되는 최대 공간량과 평균 공간량을 비교할 때 WISP은 Copy-on-write 방식에 비해 평균 19.8~47.3%, 최대 0.8~47.3% 의 공간절감을 나타내었다. 기존 데이터에 대한 업데이트 연산이 많은 Web Server 트레이스에서 WISP이 가장 큰 성능향상을 나타내었으며, 대용량 파일삭제 연산이 많은 Video Server에서는 WISP의 효과가 다른 워크로드에 비해 작게 나타났다.

3. 결 론

본 논문에서는 SCM 스토리지를 위한 새로운 스냅샷 지원 기법을 제시했다. SCM의 임의 바이트 단위의 접근이 가능하다는 특성을 활용하여 최소의 쓰기량으로 스냅샷을 지원함으로써 공간 효율성과 성능을 향상시켰다. 기존 하드디스크에서 탐색시간을 줄이기 위해 채택되었던 Copy-on-write의 방식을 제자리 쓰기 기반의 방식으로 개선하여 재귀적인 패스노드 업데이트를 제거하였다. 스냅샷 지원은 역방향 Copy-on-write방법을 통해 이루어진다. 이러한 방식은 스냅샷에 대한 접근시간이 지연된다는 단점이 있으나 스냅샷 파일시스템에 대한 접근은 빈번히 일어나지 않기 때문에 전체적인 파일시스템의 성능은 향상된다고 할 수 있다. 본 논문은 SCM 환경을 위한 스냅샷 파일시스템의 초기설계를 제시하고 있기 때문에 성능개선을 위한 추가적인 연구가 필요하다. 효율적인 스냅샷 복구 방식 및 병행적인 스냅샷 데이터에 대한 접근 방법도 연구될 필요가 있다.

참고 문헌

[1] A.-I. A. Wang, G. Kuenning, P. Reiher, and G. Popek. "The Conquest File System: Better Performance Through a Disk/Persistent-RAM Hybrid Design." ACM Transactions on Storage, Vol. 2, No. 3, pp. 309-348, 2006.

[2] R. Freitas, W. Wilcke, B. Kurdi, G. Burr, "Storage Class Memories," Tutorial at the 7th USENIX Conference on File and Storage Technologies, 2009.