재사용 거리 기반의 고성능 멀티레벨 버퍼캐시 알고리즘

박세진 ○ 박찬익
포항공과대학교
{baksejin○, cipark}@postech.ac.kr

Reuse distance based high performance multi-level buffer cache algorithm

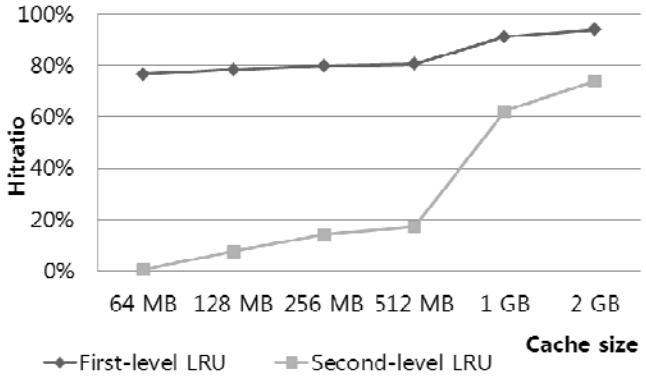
Sejin Park○ Chanik Park
POSTECH

요 약

멀티레벨 스토리지 연결 기술은 네트워크 스토리지, 클라우드 서비스, 가상화 기술 등 다양한 분야에 걸쳐 사용되고 있다. 일반적으로 멀티레벨 스토리지 연결을 효과적으로 지원하기 위해 각 레벨에서 버퍼캐시를 유지하고 있는데, 이 버퍼캐시들은 멀티레벨에서의 워크로드 특징을 효과적으로 반영하지 않고 동작하고 있다. 멀티레벨 버퍼캐시로 요청되는 블록은 최종 응용프로그램에서 직접 요청한 것이 아니라, 상위 레벨의 버퍼캐시의 캐시미스에 의해 발생된 것으로, 각 요청 블록들의 재사용 거리를 통해 워크로드의 특징을 구분 지을 수 있다. 본 논문에서는 이러한 멀티레벨 버퍼캐시의 워크로드의 특징을 살펴보고, 이를 효과적으로 지원하기 위한 재사용 거리 기반의 버퍼캐시 알고리즘을 제안한다.

1. 서 론

버퍼캐시는 요구되는 디스크 블록에 대한 디스크의 직접 입출력 요청 또는 네트워크 입출력 요청을 줄이기 위해서, 자주 사용되는 블록을 유지하고 있다. 이러한 버퍼캐시는 계층적으로 유지되는 경우가 많은데, 가상화 환경, 네트워크 스토리지 환경, 클라우드 서비스 등이 전형적인 예가 된다. 이러한 계층적 구조의 버퍼 캐시는 멀티레벨 버퍼캐시로 일컬어지며, 각 레벨에서 독립적인 블록들을 유지하게 된다. 응용 프로그램이 직접 사용하는 1차 레벨 버퍼캐시의 경우, LRU 캐시 알고리즘 등의 널리 알려져 있는 캐시 알고리즘을 통해 성능 향상을 지원하고 있으나, 2차 레벨 이상의 버퍼캐시의 경우는 LRU 알고리즘이 정상적으로 작동하지 못하는 문제를 보인다. 이는 2차 레벨 이상의 버퍼캐시의 워크로드가 1차 레벨과 다른 특성을 보이기 때문인데, 2차 레벨 이상의 버퍼캐시는 응용 프로그램이 직접 요구하는 블록이 아닌, 상위 레벨 캐시의 캐시 미스에 의해 발생한 워크로드 이다. 즉, 자주 사용 하는 블록들은 상위 레벨의 캐시에 머물러 있고, 이 상위 레벨 캐시의 크기를 벗어난 영역에 대해서 하위 레벨 캐시로 입출력을 요청하게 된다. 그림 1은 1차 레벨 캐시와 2차레벨 캐시의 캐시적중률을 비교하고 있다. 사용한 워크로드는 Microsoft Research Cambridge에서 공개한 usr0 워크로드[1] 이며, 2차레벨 캐시 적중률 그래프는 1차레벨 캐시가 64 MB 크기일 때의 결과이다.



<그림 1> 1차 레벨 LRU캐시와 과 2차 레벨 LRU 캐시의 캐시 적중률 분포도. 1차 레벨 캐시의 캐시 적중률은 75%~90%의 높은 결과를 보이지만, 2차 레벨 캐시는 64 MB ~ 512 MB 구간에서 낮은 적중률을 보이고 있음.

이렇게 캐시 적중률의 차이가 나는 이유는 2차 레벨 캐시에서 LRU 알고리즘을 사용하는 것에서 기인한다. 즉, LRU 알고리즘의 특성상, 최근에 사용된 순서대로 캐시에 유지가 되지만, 캐시에 유지된 블록들이 다시 사용될 만큼 캐시가 충분히 크지 않다. 그림 1에서 2차레벨 캐시크기가 512 MB 까지도 낮은 캐시 적중률을 보이다가 1 GB 크기를 가지자 비로서 높은 캐시 적중률을 보이기 시작한다. 이는 무조건 최근

사용순서로 캐시에 저장하는 LRU의 특성 때문이며, 본 논문에서는 이를 해결하기 위한 새로운 알고리즘을 제안한다.

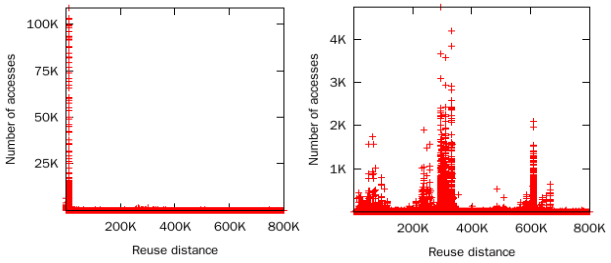
2. 기초 지식 및 관련 연구

멀티레벨 버퍼캐시에 대한 연구는 현재 많지 않으며, 대표적인 것으로 Multi-Queue (MQ) 알고리즘[2.3] 이 있다. 이 알고리즘 역시 2차 레벨의 워크로드 특성을 반영하기 위해 여러 개의 LRU 큐를 사용 빈도에 따라 순위를 정해 유지시킨다. 이 알고리즘에서는 시간적 지역성을 확보하기 위해서, 각 캐시 엔트리들이 만료시간이라는 값을 유지하는데, 이를 확인하기 위해서는 모든 LRU 큐를 다 순회해야 하는 단점을 가진다. LIRS[6] 알고리즘은 각 엔트리의 재 사용 거리를 기반으로 동작하고 있지만, 멀티레벨의 특성을 반영하지 않고, 재사용 거리가 짧은 블록 순으로 캐시를 유지 시킨다. 연구[5] 에서는 각 레벨에서 효과적으로 캐시를 유지시키기 위해서, DEMOTE라는 새로운 SCSI 명령어를 제안하기도 했으며, 파일 시스템의 의미적 분석을 통해 각 레벨의 캐시를 효과적으로 유지시키는 XRAY[4] 와 같은 연구도 진행되었다. 이 기법은 캐시는 효과적으로 유지되지만, 각 파일 시스템의 inode 정보를 매번 확인해야 하는 큰 성능적인 오버헤드를 내포하고 있다.

3. 설계

3.1 2차 캐시의 워크로드 특성

2차 캐시의 워크로드 특성은 블록 재사용 거리를 통해 나타난다. 블록 재사용 거리란, 각 블록의 사용 요청 순서 리스트에서 특정 블록이 사용된 시점부터 다시 사용된 시점까지 사이에 있는 중복을 허용하지 않는 다른 블록의 개수를 의미 한다. 예를 들어, 0-3-4-5-6-4-3-2 의 순서로 블록 사용 요청이 들어 왔을 때 블록 3 의 재사용 거리는 3 이 된다. 그림 2는 MSR-Cambridge usr0 워크로드를 구동했을 경우 1차 캐시와, 2차 캐시의 블록 재사용 거리를 나타낸 그래프이다. 2차 레벨 캐시는 1차 레벨에서 동일한 워크로드를 64 MB



(a) 1 차캐시 (b) 2 차캐시

그림 2. 1 차 캐시와 2 차 캐시의 블록 재사용 거리 그래프

크기를 가진 LRU 캐시에서 구동했다. (a) 는 1차 레벨 캐시의 블록 재사용 거리를 나타내고 있는데, 굉장히 짧은 재사용 거리를 가진 요청이 많은 것을 알 수 있으며, (b) 는 2차 레벨 캐시의 재사용 거리를 나타낸 것으로 재사용 거리 구간이 300K정도 되는 요청의 빈도가 가장 높음을 알 수 있으며, 600K정도에서도 많은 요청이 일어나고 있음을 확인할 수 있다. 300K 의 재 사용 거리의 경우, 특정 블록이 1회 사용 요청 이후, 블록의 크기가 4KB 일 경우, 약 1 GB 크기의 요청이 지난 후에 다시 사용되며, 이는 캐시 크기가 1GB 가 되지 않을 경우, 이 구간에서의 캐시 적중률은 기대하기 어렵다. 이는 그림 1의 그래프에서 나타나고 있다. 또한, 표 1은 전체 워크로드 구간 중, 한번만 접근된 (재사용 되지 않은) 블록의 비율을 나타낸 것으로 재사용 되지 않는 블록이 굉장히 많음을 확인 할 수 있다. (MSR-C의 usr0, proj0, proj3 워크로드)

<표 1> 2 차캐시에서 재사용 되지 않은 블록의 비율 – 1 차 캐시에서의 재사용 되지 않은 블록의 비율은 MSR-C 의 usr0 워크로드의 경우 6%, proj_0 경우 19%, proj_3 의 경우 29%를 보였음

First-level Workload	First-level cache size	Percentage
MSR-C usr_0	128 MB	28%
MSR-C usr_0	256 MB	30%
MSR-C usr_0	512 MB	31%
MSR-C usr_0	1024 MB	88%
MSR-C proj_0	1024 MB	68%
MSR-C proj_3	1024 MB	65%

이러한 특징은 일반적인 2차레벨 캐시에서 보여지고 있는 현상이며, 이를 위한 새로운 캐시 알고리즘이 필요하다. 2차 캐시의 특징은 다음과 같이 요약될 수 있으며, 이는 알고리즘 설계의 기반이 된다.

1) 특별히 높은 사용 빈도를 보이는 재사용 거리 구간이 존재 한다.
2) 이러한 재 사용 구간이 길 경우 해당 블록이 캐시 크기 내에서 재 사용된다는 보장이 없다
3) 자주 사용 되지 않는 블록은 캐시에 유지시키지 않고, 바로 상위 레벨 캐시로 전송할 수 있다.
4) 상위 레벨의 워크로드에 따라 특별히 높은 사용 빈도를 보이는 재사용 거리 구간이 복수 개로 나타날 수 있다.

3.2 캐시 설계 목표

2차 캐시의 워크로드 특징을 통해, 새로운 알고리즘의 설계목표는 다음과 같다.

1) 사용 빈도가 높은 특정 재사용 거리 구간을 선별적으로 캐싱
2) 접근된 블록은 재 사용 거리를 기록하기 위해 유지시킴
3) 일단 캐싱된 블록은 히트가 날 수 있는 재 사용 거리가 될 때 까지 캐시에서 유지시켜줌
4) 캐시가 가득 찼을 경우, 처음 접근된 블록은 캐싱하지 않음

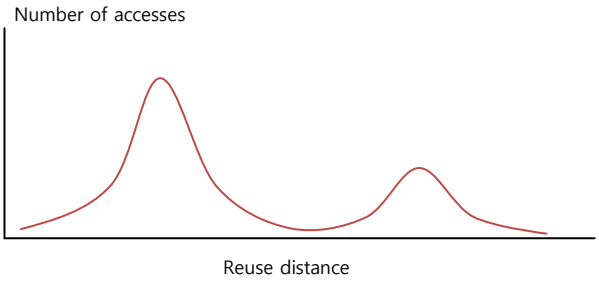
3.3 세부 캐시 동작 설명

본 연구에서 제시하는 멀티레벨 버퍼캐시는 전체 재사용 구간을 몇 개의 재 사용 구간 셋으로 나누고, 재사용 구간 셋 중 접근 빈도가 높은 셋을 Hot 셋으로 정의 한다. 그래서 현재 접근된 블록이 Hot 셋에 포함되는 블록일 경우, 이 블록을 캐싱하고 해당 Hot 셋을 만료 셋으로 기록해둔다. 속하지 않을 경우, 메타데이터만 유지시킨다. 즉, 실제 블록을 위한 메모리 할당 없이 블록의 논리 주소번호 (LBN) 값만 유지 한다. 이는 각 블록들의 재 사용 거리를 추적하기 위함이다. 본 연구에서 캐시 엔트리는 *데이터 블록 할당이 일어나는 일반 엔트리와, 메타데이터만 유지하는 그림자 엔트리의* 두 종류가 있으며, 하나의 재사용 구간 셋은 하나의 LRU 스택으로 유지된다. LRU 스택이 재사용 거리를 측정하는데 사용 되는 이유는, 엔트리가 들어오는 순으로 정렬이 되어 있기 때문에, 현재 요청이 들어오는 엔트리가 LRU 캐시 내에 있을 경우, 그 위치가 재 사용 거리가 되기 때문이다.

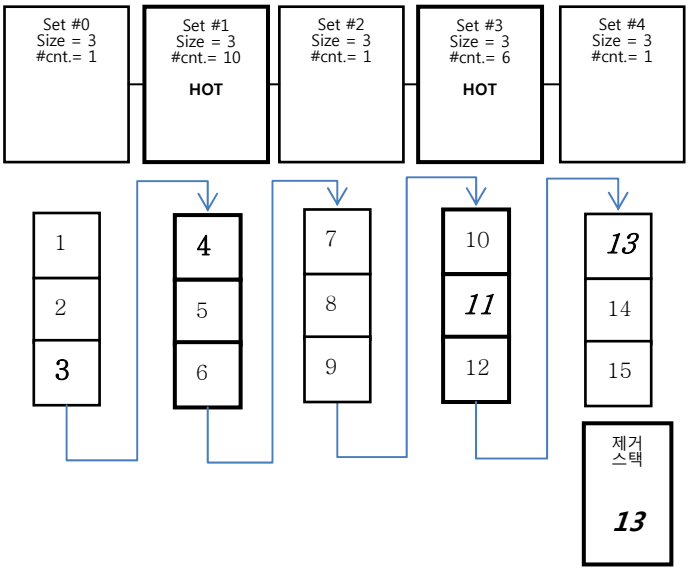
전체 구조는 그림 3과 같다. 그림 3의 (a) 와 같은 재사용 거리 그래프를 가지는 워크로드의 경우, 그림 3의 (b)와 같이 5개의 LRU 스택을 통해 효과적으로 캐시를 지원할 수 있다. 이 각 스택은 스택의 크기, 스택의 접근 누적 카운터 값을 가지고 있다. *이 누적 카운터의 값이 전체의 누적 카운터의 평균보다 높으면 Hot 셋*으로 정의 된다. 그림 3 (b)의 경우 set #1 과 set #3이 Hot 셋으로 정의가 된다. 즉, *자주 접근되는 재 사용 거리를 가진 엔트리에 대해서 선별적으로 캐싱*을 할 수 있게 된다.

앞서 설명 한 것처럼, 각 LRU 스택에는 일반 엔트리와 그림자 엔트리가 위치하고 있으며 각 스택은 전체가 하나의 LRU 스택의 순서를 유지하게 된다. 즉, set#0의 마지막 엔트리의 다음 위치는 set#1 의 첫 번째 위치로 들어간다.

그림 3의 3번, 4번 엔트리의 경우 일반 엔트리로 만료 셋이 #1 이며, 11번, 13번 엔트리의 경우 일반 엔트리로 만료 셋이 #3인 경우이다. 즉, 3번, 4번의 경우, 가장 최근에 접근되었을 때 재 사용 셋 #1에 위치 하고 있었고, 11번, 13번 엔트리의 경우 가장 최근에 접근 되었을 때 재사용 셋 #3에 위치하고 있었다. 또한 엔트리 13의 경우, 제거 스택에도 포함되어 있는데, *제거 스택에는 다음 번에 캐시에 새로운 일반 엔트리가 들어올 경우, 교체되어 나갈 엔트리가 위치*하고 있어서, 캐시 교체 시 별도의 추가 연산이 필요 없도록 해준다. 만약 *캐시가 가득 찬 상태에서, 제거 스택에 엔트리가 없을 경우, 새로운 일반 엔트리는 그림자 엔트리로 바뀌어 저장*이 된다. 이는 기존에 유지되고 있는 엔트리들이 적어도 만료 셋 까지는 캐시 내에서 유지시켜주기 위함이다. 캐시에 존재하는 일반 엔트리가 자신의 만료 셋을 지나가면 제거 스택에 추가가 된다.



(a) 재 사용 거리 그래프



(b) 캐시 크기가 4 이고, 5 개의 재사용 구간 셋을 위한 5 개의 LRU 스택과 1 개의 제거 스택으로 구성. 굵은 글씨체는 일반 엔트리, 작은 글씨체는 그림자 엔트리이며, 엔트리 3,4 는 만료 셋이 #1, 11,13 은 만료 셋이 #3. 제거 스택에는 만료 셋을 넘어선 엔트리 13 이 위치하고 있음.

그림 3. 전체 구조.

3.4 구현

본 연구에서 제안하는 알고리즘은 LRU 스택에 최소의 추가 연산만을 통해 쉽게 구현이 가능하여, O(1)의 시간 복잡도를 가지는 캐시 교체를 할 수 있다.

4. 평가

제안한 캐시 알고리즘의 평가를 위해 그림 4와 같은

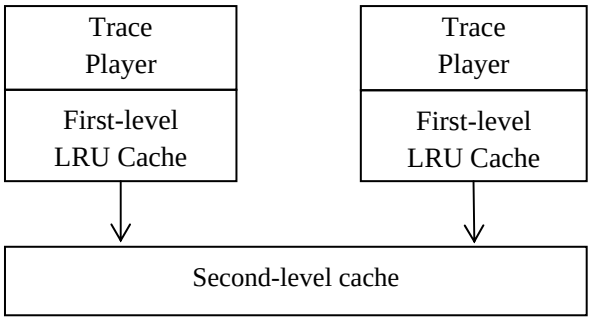


그림 4. 2차 레벨 캐시 시뮬레이터

트레이스 기반 캐시 시뮬레이터를 개발하였다. 이는 복수개의 클라이언트가 동시에 공유 캐시를 사용하는 가상화 환경, 네트워크 공유 스토리지 환경에 대한 캐시 성능 실험을 할 수 있으며, 사용한 트레이스는 MSR-C usr0와 MSR-C proj0 를 사용하였다. 그림 5~7의 실험 결과, 다양한 워크로드에서 제시한 알고리즘이 기존 LRU대비 2배 이상 성능이 좋게 나옴을 확인할 수 있었고, 특히, 그림 6의 결과와 같이 상위 레벨에서 복수개의 클라이언트가 존재하는 경우, 그 성능 차이는 더욱 커지는 것을 확인할 수 있었다.

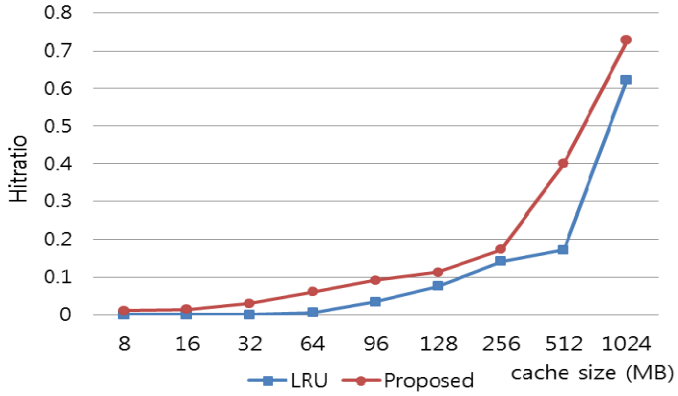


그림 5. 1차 캐시 워크로드: 1개의 64 MB LRU 캐시에서 MSR-C usr0 구동

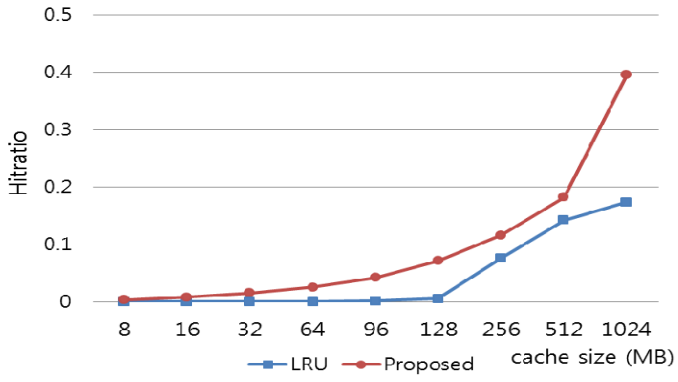


그림 6. 1차 캐시 워크로드 : 2개의 64 MB LRU 캐시에서 각각 MSR-C usr0 구동

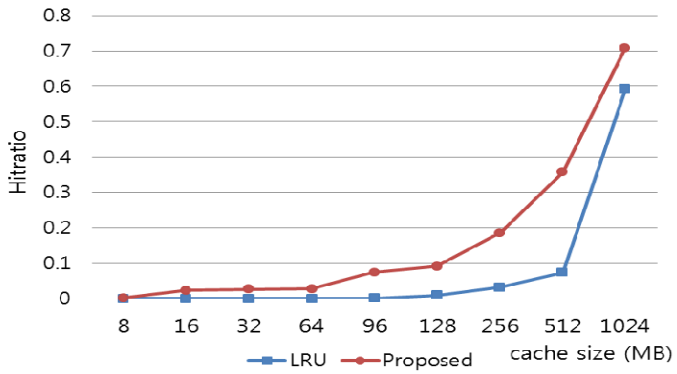


그림 7. 1차 캐시 워크로드 : 1개의 128 MB LRU 캐시에서 MSR-C proj0 구동

5. 결 론

본 연구에서는 멀티레벨 버퍼캐시에 요청되는 워크로드의 재 사용 거리가 멀다는 특성을 이용해 새로운 캐시 기법을 제안하였다. 현재 사용되고 있는 LRU보다 평균 2배 정도의 캐시 적중률을 보여주고 있다. 또한, LRU스택 기반으로 O(1)의 시간 복잡도를 가진 간단한 구현이 가능하다는 장점을 가진다.

6. Acknowledgement

이 논문은 2011년도 정부(교육과학기술부)의 재원으로 한국연구재단의 지원을 받아 수행된 연구임(No. 2011-0016972)

<참고문헌>

[1] Dushyanth Narayanan , Austin Donnelly , Antony Rowstron, Write off-loading: Practical power management for enterprise storage, ACM Transactions on Storage (TOS), v.4 n.3, p.1-23, November 2008

[2] Y. Zhou, Z. Chen, and K. Li, “Second-level Buffer CacheManagement,” IEEE Transactions on Parallel and Distributed Systems, Vol. 15, No. 7, July, 2004.

[3] Y. Zhou, J.F. Philbin, and K. Li, “Second-level Buffer Cache Management,” In Procedings of the 2001 USENIX Annual Technical Conference, 2001.

[4] L.N. Bairavasundaram, M. Sivathanu ,A.C. Arpaci-Dusseau, R.H. Arpaci-Dusseau, “X-RAY: A Non-Invasive Exclusive Caching Mechanism for RAIDs,”Proceedings of the 31st annual international symposium on Computer architecture, p.176, June 19-23, 2004.

[5] T.M. Wong and J. Wilkes, “My Cache or Yours? Making Storage More Exclusive,” In Proceedings of the USENIX Annual Technical Conference, 2002.

[6] S. Jiang and X. Zhang, “LIRS: An efficient low inter-reference recency set replacement policy to improve buffer cache performance,” in Proc. ACM SIGMETRICS Conf., 2002.