머신러닝 기반 캐시 교체 정책 성능 향상 연구

김동현, 이요셉, 나인호, 박상오*
중앙대학교, 중앙대학교, 군산대학교, *중앙대학교

dhkim@cslab.cau.ac.kr, yslee@cslab.cau.ac.kr, ihra@kunsan.ac.kr, *sopark@cau.ac.kr

A Study on the Enhancing Cache Replacement Policy with Machine Learning

Kim Dong Hyeon, Lee Yo Seb, Ra In ho, Park Sang Oh*
Chung-Ang Univ., Chung-Ang Univ., Kunsan National Univ., *Chung-Ang Univ.

요 약

전통적인 캐시 교체 전략인 LRU, MRU, LFU 는 간단하고 널리 사용되며 대부분의 경우 잘 작동하지만, 동적이고 복잡한 데이터 접근 패턴을 효과적으로 처리하지 못한다는 한계가 있다. 본 연구에서는 머신러닝 기반의 캐시 교체 정책을 활용하여 이러한 한계를 극복했다. YCSB 를 통해 다양한 데이터 접근 패턴 데이터를 생성하여 RandomForestRegressor 모델을 학습시켰다. 이를 통해 캐시 접근 패턴을 학습하여 보다 정교한 캐시 교체 결정을 가능하게 했고, 결과적으로 기존 캐시 교체 전략과 비교하여 더 높은 캐시 히트율을 달성했으며, LRU 에 비해 약 56% 높은 성능을 보여주었다. 실험 결과 머신러닝 기반의 캐시 교체 전략이 기존 전략에 비해 더 높은 성능을 제공할 수 있음을 보였다. 이는 캐시 시스템 설계 및 최적화에 새로운 방향을 제시하며, 캐시 교체 정책 분야에서 머신러닝 적용의 기초가 될 것이다.

Ⅰ. 서 론

캐시 메모리는 컴퓨터 시스템에서 중추적인 구성 요소로 데이터 액세스 시간을 크게 줄이고 전반적인 성능을 향상시킨다. 방대한 양의 데이터를 빠르게 처리해야 하는 최신 컴퓨팅 분야에서 캐시의 효율성은 매우 중요하다. [1]

하지만 캐시에 사용되는 저장장치는 기본 혹은 보조 저장장치(HDD, SSD, Flash Memory 등)에 비해 비용이 높다. 캐시 메모리의 크기를 확장하려면 상당한 비용이 발생하므로, 제한된 리소스 안에서 캐시를 효율적으로 관리하는 것이 중요한 과제이다.

캐시 성능을 판단하는 주요 지표인 hit rate 는 총 데이터 액세스 시도 횟수 대비 캐시에서 데이터를 찾은 횟수의 비율을 나타낸다. 이러한 hit rate 를 높이기 위해 효율적인 캐시 교체 정책이 필요하다.[2]

캐시 교체 정책은 캐시를 관리하는데 사용되는 전략이다. 캐시로 사용하는 공간이 가득차면 캐시 되어있는 기존 데이터들 중 일부를 제거하여 공간을 확보해야 한다. 이 때 어떤 데이터를 제거할 것인지 결정하는 것이 캐시 교체 정책이다.

본 논문에서는 머신러닝 기법을 적용한 캐시 교체 정책에 대한 새로운 접근 방식을 살펴본다. 가장 예전에 접근된 데이터(LRU), 가장 최근에 접근된 데이터(MRU), 가장 많이 접근된 데이터(LFU)를 교체하는 전통적인 캐시 교체 전략들은 단순함과 준수한 성능으로 인해 널리 채택되어 사용되어왔다. 다양한 서비스가 생겨나며 다양한 워크로드가 발생하는 요즘 시대에 전통적인 캐시 교체 전략들로는 부족한 경우가 많다.[2, 3]

2 장에서 본 논문이 제안하는 머신러닝을 활용한 보다 지능적이고 효율적인 캐시 교체 전략을 소개하고, 학습에 있어 사용한 데이터 속성과 그러한 데이터 속성을 선택한 이유를 설명한다. 3 장에서는 캐시 시스템을 재현하여 제안하는 캐시 교체 전략의 성능을 실험 및 평가한다. 4 장 결론으로 마무리한다.

Ⅱ. 본론

본 연구에서 학습에 사용된 데이터는 Yahoo! Cloud Serving Benchmark(YCSB)[4]를 통해 생성되었다. YCSB 는 클라우드 서비스 환경에서 데이터 저장 시스템의 성능을 평가하기 위해 개발된 벤치마킹 도구이다. YCSB 는 다양한 데이터 모델과 접근 패턴을 시뮬레이션하여 실제 클라우드 환경에서의 데이터베이스 성능을 측정할 수 있도록 설계되었다. 사용자가 입력한 설정을 기반으로 읽기, 쓰기, 업데이트 등 다양한 작업을 다양한 비율로 수행한다.

YCSB 를 통해 생성된 데이터는 각 I/O 에 대하여 접근 유형(읽기 또는 쓰기), 접근 시간(나노 초 타임스탬프), Key(파일 이름) 정보를 포함한다. 데이터 전처리를 통해 I/O 작업의 시퀀스에 따라 각 파일에 대한 통계 정보를 동적으로 업데이트하고 기록하도록 했다. 이를 통해 각 I/O 의 맥락에서 파일의 상태를 정확하게 파악하여 접근 패턴을 더욱 정확히 이해할 수 있도록 했다.

전처리를 거치게 되면 학습에 사용할 최종 데이터가 생성된다. 이 데이터는 Key, 읽기 빈도, 쓰기 빈도, 읽기 지역성, 쓰기 지역성, 해당 파일에 대한 최근 작업과 시간 차이, 해당 파일에 대한 최근 작업과 작업 수 차이로 구성된다.

Key 는 접근한 파일의 이름, 읽기/쓰기 빈도는 접근한 파일에 읽기/쓰기 한 횟수, 읽기/쓰기 지역성은 접근한 파일에 대한 읽기/쓰기 작업 지역성, 시간 차이는 접근한 파일의 가장 최근 작업과 시간 차이, 작업 차이는 접근한

파일의 가장 최근 작업과 현재 작업 사이에 있었던 작업의 수이다.

LFU 의 특성을 가져오기 위해 빈도를 사용했고, LRU 의 특성을 가져오기 위해 가장 최근 작업과 시간 차이와 가장 최근 작업과 현재 작업 사이에 있었던 작업의 수를 사용했다. 폭발적으로 접근이 증가하거나 지속적인 접근이 있는 등 전통적인 캐시 관리 정책은 적절하게 대응하기 어려운 워크로드에 대한 대처를 위해 지역성이라는 속성을 도입했다. 작업 지역성은 최근 접근에 더 높은 값을 부여하는 가중 이동 평균 방식으로 계산되어 접근 패턴에 기반하여 반응할 수 있도록 한다. 특정 k 시점에서 key 파일의 지역성 $Loc_k(key)$ 는 아래처럼 정의된다.

$$Loc_k(key) = \begin{cases} Loc_{k-1}(key) \times 0.6 + 0.4, & if\ accessed, \\ Loc_{k-1}(key) \times 0.6, & otherwise. \end{cases}$$

캐시된 데이터들 중 삭제할 데이터를 선택하는 과정에서 캐시 필요성이 제일 낮은 것을 삭제해야 한다. 캐시 필요성에 대한 확률을 얻어 다른 것들과 비교할 수 있도록 하기 위하여 Classifier 가 아닌 Regressor 를 선택했다. 또한 컴퓨팅 리소스가 풍부하지 않은 환경에서도 병렬 처리를 이용하여 작동할 수 있도록 하기 위해서 각각 간단한 트리들의 앙상블 기법인 RandomForest 기법을 선택했다.

모델 학습에 있어 5-fold 교차 검증을 사용하였고, 추가로 GridSearch 를 통해 최적의 하이퍼파라미터를 탐색함으로써 모델의 성능을 최적화했다.

Ⅲ. 실험

기존에 가장 전통적이며 널리 사용되고 있는 LRU, MRU, LFU 캐시 교체 정책들과 비교하였다. 성능 평가 지표로는 전체 I/O 중, 캐시 히트 횟수를 사용하였다.

본 연구의 실험은 YCSB 로 생성한 테스트용 데이터를 사용하였으며, 실제 시스템에서 발생하는 것과 비슷한 유형의 워크로드를 가진다. 테스트에 사용한 데이터셋은 10,000 번의 I/O 작업으로 구성되어 있고, Zipfian 분포를 갖는다.

테스트를 위하여 사용자 정의 캐시 시뮬레이터를 개발하여 실험을 수행했다. 이 시뮬레이터는 다양한 캐시 교체 정책으로 실제 시스템 캐시 작동을 모방한다. 시뮬레이터는 작동 중에 캐시 히트 횟수를 측정하여 각 캐시 교체 정책의 성능을 평가할 수 있도록 한다.

캐시 교체 정책	Hit Count
LRU	1033
MRU	458
LFU	962
ML based	**1611**

표 1. LRU, MRU, LFU, ML based 에 대한 실험 결과

기존 전략 중 가장 성능이 좋은 LRU는 총 10,000 번의 I/O 중, 1033 번의 Cache hit 를 달성했다. LFU 도 962 라는 준수한 수준의 성능을 보였지만, MRU 는 실험에 사용한 워크로드에 맞지 않아 458 이라는 낮은 성능을 보였다. 그에 반해 본 연구에서 제안한 머신러닝 기반의 캐시 교체 정책은 1611 이라는 높은 Hit count 를 보였다. 이는 기존 정책 중 가장 좋은 LRU 에 비해 약 56% 높은 성능이며, MRU 에 비해서는 약 252% 높은 성능이다.

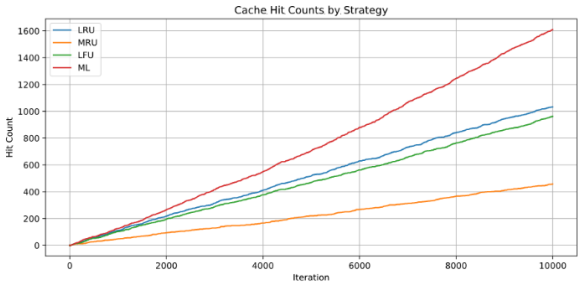

그림 1. I/O 진행에 따른 Cache Hit 횟수

Ⅳ. 결론

이 연구에서 캐시 교체 정책에 머신러닝을 적용하는 시도를 통해 기존의 정적 캐시 교체 방법론을 뛰어넘는 상당한 진전을 이루었다. 머신러닝을 적용한 캐시 교체 전략은 LRU, LFU, MRU 등의 기존 방식보다 훨씬 높은 캐시 적중률을 달성하여 캐시 관리의 효율성을 크게 향상시켰다.

이는 변화하는 액세스 패턴에 보다 효과적으로 대응할 수 있는 적응형 캐시 시스템의 가능성을 보여주었으며, 잠재적으로 캐싱 리소스를 보다 효율적으로 사용할 수 있도록 해준다.

이 연구의 가장 중요한 성과는 머신러닝 모델이 캐시 접근 패턴의 복잡성을 이해하고 기존 방법론을 뛰어넘는 교체 결정을 내릴 수 있다는 점이다. 이는 빈도나 최신성과 같은 지표 외에 그와 비슷한 복잡성을 갖는 다른 지표를 활용하여 캐시 관리에 대한 보다 정교한 접근을 가능하게 한다는 것이다.

ACKNOWLEDGMENT

이 연구는 2024 년 정부(산업통상자원부)의 재원으로 한국산업기술진흥원의 지원을 받아 수행된 연구임. (P0020632, 2024 년 산업혁신인재성장지원사업).

참 고 문 헌

[1] M. Reiss-Mirzaei, M. Ghobaei-Arani, and L. Esmaeili, 'A review on the edge caching mechanisms in the mobile edge computing: A social-aware perspective', Internet of Things, vol. 22, p. 100690, 2023.

[2] S. Kumar and P. K. Singh, "An overview of modern cache memory and performance analysis of replacement policies," 2016 IEEE International Conference on Engineering and Technology (ICETECH), Coimbatore, India, 2016, pp. 210-214.

[3] A. Jaleel, K. B. Theobald, S. C. Steely, and J. Emer, 'High performance cache replacement using re-reference interval prediction (RRIP)', in Proceedings of the 37th Annual International Symposium on Computer Architecture, Saint-Malo, France, 2010, pp. 60-71.

[4] B. F. Cooper, A. Silberstein, E. Tam, R. Ramakrishnan, and R. Sears, 'Benchmarking cloud serving systems with YCSB', in Proceedings of the 1st ACM Symposium on Cloud Computing, Indianapolis, Indiana, USA, 2010, pp. 143-154.