The Journal of The Institute of Internet, Broadcasting and Communication (IIBC)
Vol. 22, No. 1, pp.65-70, Feb. 28, 2022. pISSN 2289-0238, eISSN 2289-0246

https://doi.org/10.7236/JIIBC.2022.22.1.65
JIIBC 2022-1-10

고속 스토리지 환경의 메모리 관리를 위한 TLB 미스율 및 페이지 폴트율 모델링

Modeling of TLB Miss Rate and Page Fault Rate for Memory Management in Fast Storage Environments

박윤주*, 반효경**

Yunjoo Park*, Hyokyung Bahn**

요 약 최근 고속 스토리지의 활성화로 인해 하드디스크를 전제로 설계된 메모리 관리 시스템에 대한 재고가 필요한 시점에 이르렀다. 본 논문은 고속 스토리지 환경에서 메모리 접근 시간이 페이지 크기에 민감한 점을 관찰하고, 그 이유가 페이지 폴트율보다 TLB 미스율이 메모리 접근시간에 미치는 영향력이 커졌기 때문임을 확인하였다. 또한, TLB 미스율과 페이지 폴트율이 페이지 크기 변화에 따라 상충관계를 나타냄을 확인하고 이를 모델링하는 함수를 설계하였다. TLB 미스율의 경우 파워 피팅을 통한 모델링을 하였으며, 페이지 폴트율의 경우 2개의 항을 가진 지수 피팅을 통한 모델링을 하였다. 검증 실험을 통해 설계된 모델 함수에 의한 예측치가 실제 결과값을 잘 반영함을 확인하였다.

Abstract As fast storage has become popular, the memory management system designed for hard disks needs to be reconsidered. In this paper, we observe that memory access latency is sensitive to the page size when fast storage is adopted. We find the reason from the TLB miss rate, which has the increased impact on the memory access latency in comparison with the page fault rate, and there is trade-off between the TLB miss rate and the page fault rate as the page size is varied. To handle such situations, we model the page fault rate and the TLB miss rate accurately as a function of the page size. Specifically, we show that the power fit and the exponential fit with two terms are appropriate for fitting the TLB miss rate and the page fault rate, respectively. We validate the effectiveness of our model by comparing the estimated values from the model and real values.

Key Words : fast storage, NVM, non-volatile memory, page fault, storage

Ⅰ. 서 론

하드디스크와 메모리 사이의 큰 성능 차이로 인해 전통적인 컴퓨터 시스템의 메모리 관리는 디스크 접근을 최소화하는 것에 초점을 맞추어왔다[1, 2, 3]. 하드디스크의 접근에는 수십 밀리초 이상의 시간이 소요되며, 이는 DRAM 접근 시간의 십만배 이상에 해당한다. 한편,디스크 접근 중 가장 오랜 시간이 소요되는 부분은 헤드의 이동 시간이므로 한번의 헤드 이동으로 가능한 많은 데이터를 읽어오는 것이 효율적이다. 이와 같이 디스크의 접

*비회원, 이화여자대학교 컴퓨터공학과
**정회원, 이화여자대학교 컴퓨터공학과
접수일자 2021년 10월 4일, 수정완료 2022년 1월 8일
게재확정일자 2022년 2월 4일

Received: 4 October, 2021 / Revised: 8 January, 2022 / Accepted: 4 February, 2022
*Corresponding Author: bahn@ewha.ac.kr
Dept. of Computer Engineering, Ewha University, Korea

근이 느리다는 점과 헤드의 이동이 필요하다는 점을 감안할 때, 디스크의 입출력 단위는 가급적 크게 설정하는 것이 효율적이다. 그러나, 메모리의 전체 용량이 유한하므로, 많은 양의 데이터를 디스크에서 읽어올 경우 이미 메모리에 존재하는 데이터 중 일부를 삭제해야 하는 상황이 발생한다. 따라서, 디스크로부터 한번에 읽어오는 데이터의 크기를 늘이는 것이 효과적이기 위해서는 읽어온 데이터가 실제로 사용되거나 메모리 크기에 여유가 있어야 한다. 이러한 이유로 페이지의 크기는 보통 4KB로 한정하고, 미리읽기 옵션을 두어 추가로 읽어올 페이지의 수를 데이터의 접근 패턴에 기반해 결정한다. 예를 들어 데이터가 순차적으로 접근될 경우 미리읽기의 페이지 수를 늘리고 그렇지 않은 경우 줄인다.

한편, 최근 플래시메모리, NVM 등 고속 스토리지의 등장으로 메모리와 스토리지 사이의 성능 격차가 줄어들고 있다[4, 5, 6]. 플래시메모리의 접근 시간은 수십 마이크로초로 DRAM과의 접근시간 격차를 천배 이내로 줄였으며, 최근 NVM의 제품화로 이러한 성능 격차는 수십배 이내로 줄어들 상황에 이르렀다[7, 8]. 이에 NVM 스토리지의 효율적인 관리를 위한 많은 연구들이 이루어졌으며[5, 8], 인텔의 옵테인 등 NVM 제품군이 상용화되기에 이르렀다. 한편, NVM은 헤드의 이동이 없기 때문에 데이터 전송량이 늘수록 접근시간도 길어지는 특성이 있다. 즉, 한번의 헤드 이동으로 많은 데이터를 읽어오는 것이 효율적이던 하드디스크에서의 특성은 더 이상 유효하지 않은 상황으로 바뀌었다. 이러한 이유로 소규모 페이지를 사용하는 것이 NVM 스토리지에서는 더 효율적일 수 있다.

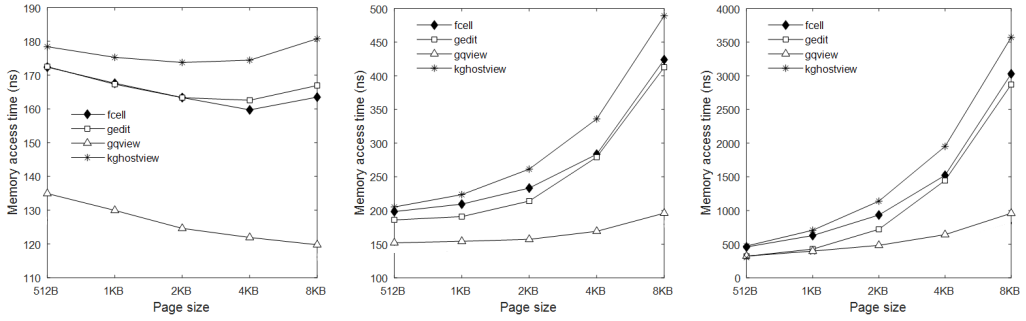
한편, NVM 스토리지의 도입으로 메모리 접근시간의 병목점이 페이지폴트 처리시간이 아닌 메모리 주소변환 시간으로 바뀔 수 있는 상황에 이르렀다. 메모리 접근을 위해서는 가상 메모리 주소를 물리적 메모리 주소로 변환하는 주소변환절차를 거친 후 실제 데이터를 메모리에서 접근하게 된다. NVM 스토리지의 도입으로 페이지 폴트가 발생하더라도 실제 데이터를 접근하는 시간이 많이 줄어 주소변환에 소요되는 시간의 비중이 상대적으로 커지게 되었다. 한편, 주소변환과정의 가속을 위해 TLB(translation lookaside buffer)라는 주소변환캐시를 두고 있는데, 이러한 TLB의 용량이 제한적이므로, TLB를 통한 주소변환이 실패할 경우 메모리 상의 주소변환 페이지테이블을 추가로 접근해야 하는 부담이 뒤따른다. 비록 페이지 크기를 줄이는 것이 데이터 접근시간 측면에서 효율적이지만, 이는 TLB의 히트율을 떨어뜨려 주소변환시간을 길어지게 한다.

즉, TLB의 성능을 높이기 위해서는 페이지 크기를 늘이는 것이 더 바람직하다. 이는 고정된 TLB 용량으로 더 많은 메모리 영역에 대한 주소변환을 커버할 수 있기 때문이다. 이와 같은 두 가지 상충되는 상황으로 인해, 데이터 접근 시간과 주소변환시간 사이의 트레이드 오프를 유심히 분석하여 적절한 페이지 크기를 결정하는 것이 필요하다. 또한, 최적의 페이지 크기는 메모리의 여유공간이 어느 정도인지에 따라서도 달라지므로, 페이지 크기를 결정하는 것은 여러 상황을 종합적으로 고려해야 하는 복잡한 문제이다.

이러한 상황에 대처하기 위해 본 논문에서는 페이지 크기 변화에 따른 주소변환시간과 데이터 접근시간의 예측을 위한 모델을 제안한다. 특히, 본 논문은 TLB 미스율과 페이지 폴트율의 모델링을 위해 각각 거듭제곱함수 피팅과 2항 기반의 지수함수 피팅을 수행하고 실제 결과값과의 비교를 통한 검증을 수행한다.

II. 선행 실험

고속 스토리지인 NVM의 접근시간은 DRAM보다 100배 이상 느린 수준에서 DRAM과 유사한 수준까지 다양하게 예측되고 있다. 본 장에서는 NVM의 성능이 변화함에 따라 메모리 접근시간이 어떻게 변하는지를 확인하는 선행실험을 하였다. 그림 1은 NVM의 접근시간이 DRAM과 동일한 경우부터 10배, 100배로 늘어남에 따라 워크로드별 평균 메모리 접근시간이 어떻게 변하는지를 페이지 크기별로 보여주고 있다. 본 논문의 실험에 사용한 워크로드는 리눅스용 응용인 freecell, gedit, gqview, kghostview의 4종이다. 그림에서 보는 것처럼 NVM의 접근시간이 DRAM의 10배에서 100배까지로 느려짐에 따라 소규모 페이지가 더 좋은 성능을 나타냄을 확인할 수 있다. 이에 비해 그림 1(a)에서 보는 것처럼 NVM 접근 시간이 DRAM에 가까워짐에 따라 최적의 페이지 크기는 워크로드의 종류에 따라 달라지는 것을 확인할 수 있다. gqview의 경우 페이지 크기가 커질수록 우수한 성능을 나타내었다. 이는 페이지폴트 처리시간이 충분이 짧은 상황에서 주소변환시간이 메모리 접근시간의 주요한 요인이 되기 때문이다. 즉, 이러한 경우에는 TLB 히트율이 페이지 폴트율보다 메모리 성능에 더 중요한 영향을 미친다는 의미이다. 그러나, 다른 워크로드의 경우 페이지 크기가 증가함에 따라 메모리 접근시간



(a) 1x of DRAM (b) 10x of DRAM (c) 100x of DRAM

그림 1. 페이지 크기 및 스토리지 접근시간 변화에 따른 메모리 접근시간
Fig. 1. Total Memory access time as a function of page size and storage access time.

이 늘어나는 것을 확인할 수 있다. 이는 이러한 워크로드에서 지역성이 상대적으로 약해서 페이지 폴트율이 TLB 히트율보다 메모리 성능에 더 중요한 영향을 미치기 때문이다.

요약하자면 최적의 메모리 성능은 고정된 페이지 크기에서 얻어질 수 없고 워크로드 특성과 스토리지의 접근시간에 따라 달라짐을 뜻한다. 이는 고속 스토리지 환경에서 전통적인 4KB의 페이지 크기를 설정하기보다 워크로드 및 시스템 환경에 맞게 페이지 크기를 적절하게 설정하기 위한 모델이 필요함을 의미한다. 특히, 메모리 접근시간이 주소변환시간과 데이터 접근시간으로 구분되고, 양 시간요소 중 병목점이 워크로드나 시스템 환경에 따라 달라지므로 이를 잘 예측하기 위해 페이지 크기에 따른 TLB 히트율과 페이지 폴트율을 모델링하는 연구가 필요함을 의미한다. 일반적으로 데이터 접근시간은 페이지 크기가 작을수록 더 개선되는 경향이 있으나 주소변환시간은 페이지 크기가 클수록 더 개선되는 특징이 있다. 이러한 이유로 주어진 상황에서 두 시간요소 중 어느 쪽이 상대적으로 더 중요한지를 예측하고 그에 맞게 페이지 크기를 결정할 수 있는 모델이 필요하다.

III. 모델링 및 검증

가장 널리 사용되는 페이지 크기는 4KB이지만, 리눅스 등의 운영체제들은 미리읽기 옵션을 통해 요청된 페이지뿐 아니라 인접 페이지들을 최대 128개까지 동시에 읽어들일 수 있다. 이는 디스크 환경에서 데이터의 크기와 무관하게 스토리지를 접근하는 기본비용이 매우 높기 때문에 사용 가능한 옵션이다. 일부 대용량 메모리 하에서는 4MB에 이르는 거대 페이지를 사용하기도 한다. 한편, 스토리지가 빨라지고 헤드의 이동 같은 기본 비용이 거의 없는 고속 스토리지 환경에서는 페이지 크기를 늘이는 것이 효과적인 상황은 더 이상 아니라고 볼 수 있다.

그러나, 페이지 크기를 줄이는 것은 주소변환시간을 늘어나게 하는 부작용을 초래한다. 이는 페이지 크기가 작을수록 TLB에 의한 메모리 주소변환 가능성이 줄어들기 때문이다. 따라서, 최적의 페이지 크기를 결정하기 위해서는 주소변환비용과 스토리지 접근비용 간의 상충관계를 고려해야 한다. 페이지 크기가 메모리 접근시간에 미치는 영향을 분석하기 위해 본 논문은 페이지 크기에 따른 TLB 히트율과 페이지 폴트율을 예측할 수 있는 모델을 설계한다.

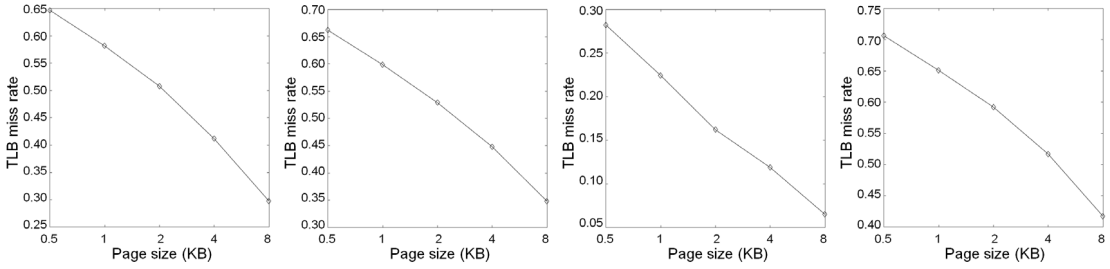
메모리 접근을 위한 주소변환은 메모리 상에 존재하는 페이지 테이블을 통해 이루어지는데, 이러한 주소변환의 성능을 개선하기 위해 페이지 테이블의 일부를 메모리보다 빠른 TLB에 보관한다. 따라서, 페이지에 대한 접근요청이 발생하면, 먼저 TLB를 통한 주소변환을 시도하고, 실패한 경우 페이지 테이블을 참조한다. 주소변환이 끝나면 변환된 주소를 가지고 해당 메모리 위치의 데이터 접근이 이루어진다. 이때, 해당 페이지가 메모리에 존재하지 않아 페이지 폴트가 발생한 경우 스토리지 접근이 선행되어야 한다.

TLB 미스율을 R_T, 페이지폴트율을 R_F 라 할 때 메모리 접근시간 T는 아래와 같이 표시할 수 있다.

$$T = T_{ADDR} + T_{DATA} \quad (1)$$

$$T_{ADDR} = (1 - R_T) * t_\varepsilon + R_T * (t_\varepsilon + t_r) \quad (2)$$

$$T_{DATA} = (1 - R_F) * t_r + R_F * (t_r + t_p) \quad (3)$$



(a) freecell (b) gedit (c) gqview (d) kghostview

그림 2. 워크로드별 TLB 미스율
Fig. 2. TLB miss rate for different workloads.

이때, T_{ADDR}은 주소변환시간, T_{DATA}는 데이터 접근시간, t_ε는 TLB 접근시간, t_r는 메모리 접근시간, t_p는 페이지폴트 처리시간을 뜻한다. TLB의 엔트리 수가 고정돼 있으므로 빈 엔트리가 없을 때 새로운 주소변환정보를 저장하기 위해서는 기존 항목 중 하나를 삭제해야 한다. 삭제 대상을 선정하기 위해 본 논문은 TLB 캐시 관리에 널리 사용되는 LRU 알고리즘을 사용하였다. 한편, 메인 메모리 용량이 한정적이므로 메모리 내에 가용 페이지가 없는 경우 새로운 페이지를 담기 위해 기존 페이지 중 어느 하나를 삭제하는 정책이 필요하다. 본 논문에서는 가상메모리 시스템에서 널리 사용되는 클럭 알고리즘을 사용해서 삭제할 대상을 선정하도록 하였다[4].

그림 2는 메모리 참조 트레이스를 이용한 각 워크로드의 재현 실험을 통해 페이지 크기에 따른 TLB 미스율이 어떻게 변화하는지를 표시하였다. 그림에서 보는 것처럼 TLB 미스율은 페이지 크기가 커질수록 개선되는 것을 확인할 수 있다. 또한, 페이지 크기가 8KB에 이르는 상황에서도 곡선의 경사는 매우 가파른 것을 확인할 수 있다. 한계효용의 법칙에 의하면 TLB 미스율은 페이지 크기가 충분히 큰 경우 더 이상 개선되지 않는 지점이 발생하는 것이 타당할 것이다. 이를 확인하기 위해 kghostview 워크로드에 대해 페이지 크기를 더욱 증가시키는 추가 실험을 하여 그림 3에 x로 표시하였다. 그림에서 보는 것처럼 페이지 크기가 더욱 증가함에 따라 TLB 성능의 개선 폭이 완만해지는 한계효용을 확인할 수 있다. 본 논문에서는 TLB 성능을 예측하는 모델을 세우기 위해 페이지 크기에 따른 TLB 미스율 그래프를 피팅하였다. 다양한 피팅 방법을 시도한 결과 파워 피팅이 TLB 미스율을 가장 잘 모델링함을 확인하였으며, 그림 3은 x로 표시된 실제 TLB 미스율과 파워 피팅 결과 만들어진 모델에 의한 그래프를 비교해서 보여주고 있다. 그림에서 보는 것처럼 제안한 모델은 실제 TLB 미스율을 잘 예측함을 확인할 수 있다.

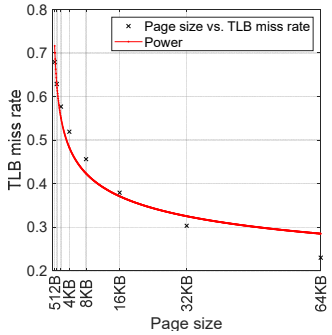
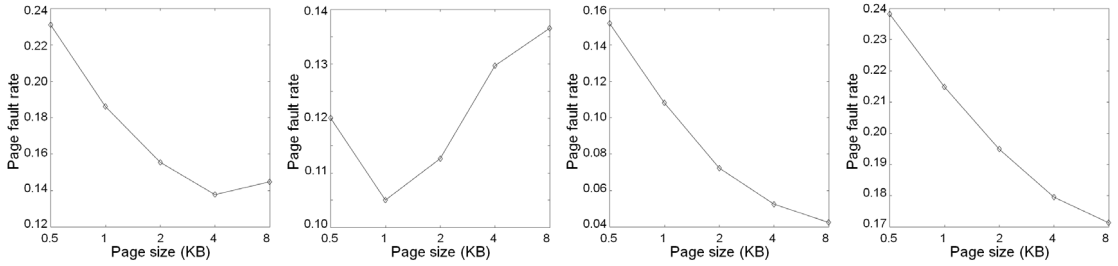


그림 3. 모델과 실제 TLB 미스율의 비교
Fig. 3. Comparison of modeled and real TLB miss rate.

이제 페이지 폴트율에 대한 예측 모델을 살펴보도록 하겠다. 페이지 폴트율은 페이지 크기뿐 아니라 워크로드 대비 메모리 용량이 어느 정도인지에도 좌우된다. 고정된 메모리 용량으로 페이지 크기를 증가시킬 경우 메모리 상에 담을 수 있는 총 페이지의 수는 줄어든다. 일반적으로 지역성의 원리에 의해 페이지 폴트율은 페이지 크기가 증가할수록 개선되는 특성이 있다. 그러나, 페이지 크기가 커지면 한정된 메모리에 담을 수 있는 데이터의 다양성이 떨어지므로 페이지 폴트율이 다시 나빠지는 지점이 존재한다. 따라서, 페이지 폴트율 곡선에는 변곡점이 존재하며 그 위치는 메모리 용량과 워크로드 특성에 따라 가변적이다. 특히, 주어진 워크로드 대비 메모리 용량이 상대적으로 작은 경우 변곡점은 더 빨리 나타나며, 이는 메모리 크기가 충분한 상황보다 최적의 페이지 크기가 더 작을 수밖에 없음을 시사한다.

그림 4는 페이지 크기 변화에 따른 워크로드별 페이지 폴트율을 보여주고 있다. 곡선의 모양이 워크로드에 따



(a) freecell (b) gedit (c) gqview (d) kghostview

그림 4. 워크로드별 페이지 폴트율
Fig. 4. Page fault rate for different workloads

라 다른 것을 확인할 수 있으며 워크로드가 동일한 경우에도 시스템 내의 메모리 용량에 따라 곡선의 모양이 달라질 수 있다. 그림 5는 메모리 크기가 워크로드 풋프린트의 30%에서 100%까지 변화함에 따라 페이지 폴트율이 어떻게 변하는지를 보여주고 있다. 그림에서 보는 것처럼 메모리 크기가 달라짐에 따라 페이지 폴트율 곡선

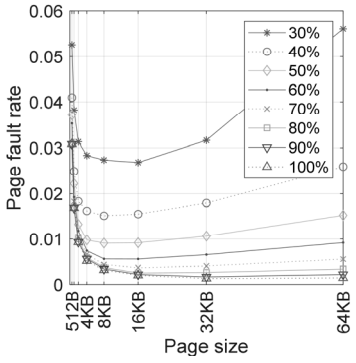


그림 5. 메모리 크기 변화에 따른 페이지 폴트율
Fig. 5. Page fault rate for different memory sizes.

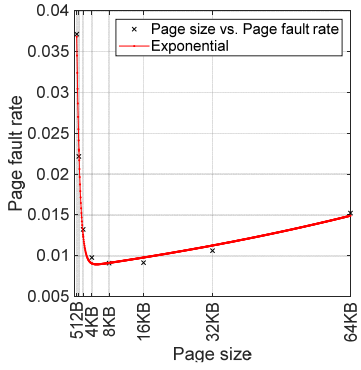


그림 6. 모델과 실제 페이지 폴트율의 비교
Fig. 6. Comparison of modeled and real page fault rate.

의 추세가 달라지는 것을 확인할 수 있다.

이와 같은 추세를 모두 반영하는 함수를 찾기 위해 본 논문에서는 다양한 시도를 하였으며, 그 결과 2개의 항을 가진 지수 피팅을 통해 페이지 폴트율을 모델링하였다. 그림 6은 메모리 크기가 50%인 경우 실제 페이지 폴트율과 본 논문이 설계한 모델에 의한 예측치를 비교해서 보여주고 있다. 그림에서 보는 것처럼 제안한 모델은 실제 페이지 폴트율을 잘 예측하는 것을 확인할 수 있으며, 이러한 상황은 메모리 크기나 워크로드 변화에도 잘 동작하는 것을 확인할 수 있었다.

Ⅳ. 결 론

본 논문에서는 NVM이 스토리지로 사용되는 시스템에서 워크로드 및 시스템의 다양한 환경 변화에 따른 메모리 접근 시간을 분석하고 이를 잘 예측하는 모델을 설계하였다. 특히, 페이지 크기가 변함에 따라 메모리 접근 시간이 어떻게 변화하는지를 예측하기 위해 페이지 폴트율과 TLB 미스율을 잘 예측할 수 있는 함수를 피팅하였다. TLB 미스율의 경우 파워피팅을 통해 예측 모델을 설계했으며, 페이지 폴트율은 2개의 항을 이용한 지수 피팅을 통해 예측 모델을 설계하였다. 설계한 모델은 실제 TLB 미스율 및 페이지 폴트율을 잘 예측할 수 있음을 확인하였다. 본 논문의 결과는 고속 스토리지가 실용화되는 시점에 메모리 및 스토리지 시스템의 효율적인 관리에 활용될 수 있을 것으로 기대된다.

References

[1] S. Ng, "Advances in disk technology: performance

issues," IEEE Computer, vol. 31, no. 5, pp. 75-81, 1998.
DOI: https://doi.org/10.1109/2.675641

[2] T. Kim and H. Bahn, "Implementation of the storage manager for an IPTV set-top box," IEEE Trans. Consumer Electronics, vol. 54, no. 4, pp. 1770-1775, 2008.
DOI: https://doi.org/10.1109/TCE.2008.4711233

[3] O. Kwon, H. Bahn, and K. Koh, "Popularity and prefix aware interval caching for multimedia streaming servers," Proc. 8th IEEE International Conference on Computer and Information Technology, pp. 555-560, 2008.
DOI: https://doi.org/10.1109/CIT.2008.4594735

[4] J. Park, H. Lee, S. Hyun, K. Koh, and H. Bahn, "A cost-aware page replacement algorithm for NAND flash based mobile embedded systems," Proc. ACM EMSOFT Conference, pp. 315-324, 2009.
DOI: https://doi.org/10.1145/1629335.1629377

[5] D. Kim, E. Lee, S. Ahn, and H. Bahn, "Improving the storage performance of smartphones through journaling in non-volatile memory," IEEE Trans. Consumer Electronics, vol. 59, no. 3, pp. 556-561, 2013.
DOI: https://doi.org/10.1109/TCE.2013.6626238

[6] S. Byun, "Search performance improvement of column-oriented flash storages using segmented compression index," Journal of the Korea Academia-Industrial cooperation Society, vol.14, no. 1, pp. 393-401, 2013.
DOI: https://doi.org/10.5762/KAIS.2013.14.1.393

[7] I. Shin, "Performance evaluation of applying shallow write in SSDs with internal cache," The Journal of KIIT, vol. 17, no. 1, pp. 31-38, 2019.
DOI: https://doi.org/10.14801/jkiit.2019.17.1.31

[8] S. Yoon, H. Park, K. Cho, and H. Bahn, "Optimization techniques for power-saving in real-time IoT systems using fast storage media," The Journal of the Institute of Internet, Broadcasting and Communication, vol. 21, no. 6, pp. 71-76, 2021.
DOI: https://doi.org/10.7236/JIIBC.2021.21.6.71

저 자 소 개

박 윤 주(비회원)

- 2015년 2월 : 이화여자대학교 컴퓨터공학과 학사
- 2015년 3월 ~ : 이화여자대학교 컴퓨터공학과 통합과정
- 주관심분야 : 운영체제, 스토리지 시스템, 임베디드 시스템

반 효 경(정회원)

- 1997년 2월 : 서울대학교 계산통계학과 학사
- 1999년 2월 : 서울대학교 전산과학과 석사
- 2002년 2월 : 서울대학교 컴퓨터공학부 박사.
- 2002년 9월 ~ : 이화여자대학교 컴퓨터공학과 교수.
- 주관심분야 : 운영체제, 스토리지시스템, 임베디드시스템

※ This work was supported by the National Research Foundation of Korea (NRF) grant funded by the Korea government (MSIT) (No. 2019R1A2C1009275) and also by the ICT R&D program of MSIT/IITP (2020-0-00121, Development of data improvement and dataset correction technology based on data quality assessment).