JKSCI

한국컴퓨터정보학회논문지
Journal of The Korea Society of Computer and Information
Vol. 25 No. 2, pp. 1-10, February 2020
https://doi.org/10.9708/jksci.2020.25.02.001

2WPR: Disk Buffer Replacement Algorithm Based on the Probability of Reference to Reduce the Number of Writes in Flash Memory

Won Ho Lee*, Jong Wook Kwak*

*Student, Dept. of Computer Engineering, Yeungnam University, Gyeongsan, Korea
*Professor, Dept. of Computer Engineering, Yeungnam University, Gyeongsan, Korea

[Abstract]

In this paper, we propose an efficient disk buffer replacement policy which improves hit ratio and reduces writing operations of flash based storages. The flash based storage has many advantages, including a small form factor, non-volatility and high reliability, but there are problems caused by own limitations, like not-in-place update, short life cycle and asymmetric I/O latencies. To redeem these problems, this paper proposes the write weighted probability of reference(2WPR) policy. 2WPR policy predicts re-referencing probability and calculates localities of each page. Furthermore, by weighting write operations to every pages, 2WPR can reduce write operations to flash based storage. In addition, we can improve the performance with higher hit ratio and reduce the number of write operations and consequently shorten the latencies of each operation. The results show that our policy provides improvements of up to 10% for the hit ratio with the reduction of up to 5% for the flash writing operation compared with other policies.

▸ **Key words**: Buffer replacement policy, Flash based storage, Localities of pages, Weighted write, Probability of reference

[요 약]

본 논문에서는 향상된 히트율과 더 적은 낸드 플래시 메모리 쓰기 연산을 할당하는 디스크 버퍼 교체 정책을 소개한다. 플래시 메모리는 높은 집적도, 높은 신뢰성 및 비휘발성이라는 특징을 가지고 있어 최근 많은 곳에서 사용되고 있다. 하지만 삭제 이후 쓰기 연산 문제, 비대칭적인 연산 속도와 짧은 수명 등의 한계점도 가지고 있다. 이런 문제를 개선하기 위해 본 논문에서는 2WPR 정책을 소개한다. 2WPR 정책은 디스크 버퍼의 각 페이지마다 이후 재참조될 가능성, 각 지역성 및 쓰기 연산에 대한 가중치 분석을 통해 교체할 페이지를 선택한다. 제안된 새로운 정책은 기존 디스크 버퍼 관리 정책에 비해 히트율을 최대 10%까지 향상시킬 수 있으며 플래시 메모리에 대한 쓰기 연산을 최대 5%까지 감소시킬 수 있었다.

▸ **주제어**: 버퍼 교체 정책, 플래시 기반 저장장치, 페이지 지역성, 쓰기 가중, 참조 가능성

• First Author: Won Ho Lee, Corresponding Author: Jong Wook Kwak
*Won Ho Lee (wonholee@ynu.ac.kr), Dept. of Computer Engineering, Yeungnam University
*Jong Wook Kwak (kwak@yu.ac.kr), Dept. of Computer Engineering, Yeungnam University
• Received: 2019. 12. 24, Revised: 2020. 01. 14, Accepted: 2020. 01. 17.

http://www.ksci.re.kr pISSN:1598-849X | eISSN:2383-9945

I. Introduction

플래시 메모리(Flash memory)는 최근 임베디드 시스템에서 각광받는 저장 장치 중 하나이다. 플래시 메모리는 전원이 끊어져도 데이터를 보존할 수 있는 비휘발성 저장 장치의 일종이며 전기적으로 자유롭게 데이터를 쓰고 지울 수 있다. 또한 낸드 플래시 메모리는 일반적인 자기 디스크에 비해 접근 속도가 빠르고 소형화 및 경량화가 쉽고 내구성이 높다는 특징을 갖는다 [1]. 이는 일반적인 저장 매체 중 하나인 자기 디스크와 달리 물리적/기계적 장치가 포함되어 있지 않기 때문이다. 이런 장점을 가지는 덕분에 플래시 메모리는 SSD(Solid State Drive)와 USB(Universal Serial Bus) 메모리, 그리고 휴대전화를 비롯한 많은 휴대용 임베디드 장치에서 이용되기 유리하다. 이처럼 풍부한 수요를 바탕으로 플래시 메모리에 대해 지속적인 연구가 이루어진 결과, 집적도가 늘어나고 가격 역시 낮아지며 사용량 또한 점차 증가하는 추세이다 [2].

플래시 메모리로 구성되는 저장 장치는 기존에 많이 이용되던 자기 디스크 기반의 저장 장치와는 다른 몇 가지 특징을 가진다. 첫 번째 특징은 연산의 종류에 따라 접근 단위가 다르며 처리 속도가 불균형하다는 점이다. 플래시 메모리는 읽기 및 쓰기 연산을 수행할 때 페이지 단위로 처리하나 삭제 연산의 경우 블록 단위로만 이루어진다. 따라서 삭제 연산은 그 속도가 다른 연산들에 비해 상대적으로 느리다. 두 번째 특징은 이미 사용한 영역에 대해 덮어쓰기 연산을 할 수 없다는 점이다. 앞서 제시된 첫 번째 특징과 맞물려 이미 데이터가 존재하는 블록에 새로운 데이터를 쓰기 위해서는 먼저 해당 블록에 저장된 데이터를 삭제하는 연산이 강제된다. 세 번째 특징은 한 블록에 대해 수행할 수 있는 삭제 연산의 횟수가 제한된다는 점이다. 이는 플래시 메모리의 수명 제한으로 이어진다. 결과적으로 플래시 메모리는 같은 자리에 대한 삭제 이후 쓰기 연산(Erase-before-write)이 많아질수록, 즉 제자리 덮어쓰기가 많아질수록 시스템의 성능이 하락하는 결과를 초래한다 [3].

많은 연구에서 이런 플래시 메모리의 태생적 문제를 해결하기 위해 디스크 버퍼를 추가하고 이를 관리할 수 있는 디스크 버퍼 관리 정책을 다루고 있다. 디스크 버퍼 관리 정책들의 공통적인 목적은 플래시 메모리에 직접적으로 할당되는 연산을 완화하여 플래시 메모리의 활용도 및 수명을 늘리는 것이다.

그러나 이전의 자기 디스크와 플래시 메모리의 특성이 완전히 다르기 때문에 자기 디스크에서 이용되는 디스크 버퍼 관리 정책을 플래시 메모리에 그대로 적용하기에는 무리가 따르며, 이 때문에 새로운 정책이 연구될 필요성이 생긴다.

본 논문에서는 이러한 필요성에 따라 새로운 디스크 버퍼 관리 정책인 2WPR(Write Weighted Probability of Reference)을 제안한다. 2WPR 정책은 디스크 버퍼에 저장되는 페이지에 대해 이후에도 참조될 가능성 및 공간 지역성을 각각 분석하며, 소거에 가장 적합한 페이지부터 플래시 메모리로 퇴거시킴을 통해 디스크 버퍼에 참조될 가능성이 더 높은 페이지들을 잔류시킨다. 그 결과, 2WPR은 디스크 버퍼를 더 효율적으로 관리하여 플래시 메모리의 수명을 향상시킴과 동시에 높은 버퍼 적중률을 보장하여 시스템의 성능 역시 향상시킨다.

본 논문의 구성은 다음과 같다. 2장에서는 플래시 메모리의 특성과 SSD, 디스크 버퍼 관리 정책에 대한 관련 연구를 소개한다. 3장에서는 더 효율적으로 디스크 버퍼를 관리하기 위해 본 연구에서 제안하는 2WPR 알고리즘의 구성 요소에 대해 구체적으로 기술한다. 4장에서는 제안하는 디스크 버퍼 관리 정책에 대한 성능을 다양한 실험 결과를 이용하여 평가한다. 마지막으로 5장에서는 결론에 대해 서술한다.

II. Background & Related works

1. NAND flash memory

낸드 플래시 메모리는 기존의 자기 디스크와 비교했을 때 매우 빠른 연산 속도와 뛰어난 내구성 등의 장점을 가지지만 단점도 가진다. 첫 번째는 연산에 걸리는 시간이 비대칭이라는 점이다. 일반적으로 읽기 연산에 소요되는 시간보다 쓰기 연산에 소요되는 시간이 더 느리다. 또한 지우기(삭제) 연산은 쓰기 연산보다 더욱 느린 속도로 처리된다. DRAM과 낸드 플래시 메모리의 연산 속도 비교는 Table 1과 같다 [4-6]. 두 번째는 제자리 덮어쓰기가 불가능하다는 점이다. 낸드 플래시 메모리는 이미 데이터가 존재하는 페이지에 대한 쓰기 연산이 발생하면 먼저 지우기 연산을 수행한 다음 쓰기 연산을 해야만 한다. 그로 인하여 제자리 덮어쓰기가 자주 발생하는 경우라면 읽기 연산 및 쓰기 연산보다 느린 지우기 연산이 자주 발생해 전체적인 시스템의 성능 및 효율이 하락한다.

Table 1. Operation Latency of DRAM and NAND

Operation	Unit	Latency	
		DRAM	NAND flash
Read	Page	15 ns	21 μs
Write	Page	15 ns	200 μs
Erase	Block	N/A	1.2 ms

Table 2. SSD Bit Cell Type and P/E Cycles

	SLC	MLC	TLC
Bit per cell	1	2	3
P/E cycles	10^5	10^4	10^3

세 번째는 낸드 플래시 메모리의 수명 문제이다. 낸드 플래시 메모리는 비트를 저장할 수 있는 셀로 구성되어 있다. 각 셀은 플로팅 게이트에 전자를 저장하고 빼내는 방법으로 데이터를 저장하는데, 이 때 강한 전압을 절연체에 통과시킨다. 그러나 전압을 통과시킬 때마다 절연체에 조금씩 손상이 가해져 통과시키는 전압이 불안정해진다. 결국 낸드 플래시 메모리는 일정한 수명을 가지게 된다. Table 2는 플로팅 게이트 내에 존재하는 전자들의 양에 따라 구분되는 낸드 플래시 메모리의 종류와 특성을 보여준다 [7-10].

2. Disk buffer policy

디스크 버퍼는 호스트 시스템에서 발생하는 쓰기나 읽기 연산에 대한 요청을 낸드 플래시 메모리에 할당하지 않고 대신 흡수하는 공간이다. 디스크 버퍼에서 관리되는 데이터가 많을수록 낸드 플래시 메모리에 직접 발생하는 요청이 줄어든다. 이를 통해 읽기 및 쓰기에 대한 성능이 향상되고 낸드 플래시 메모리가 가지는 수명 문제나 쓰기 연산에서 발생하는 오버헤드를 상쇄할 수 있다. 그러나 디스크 버퍼의 크기는 낸드 플래시 메모리보다 더 작기 때문에 항상 필요한 데이터를 가지고 있기란 불가능에 가깝다. 이러한 이유로 디스크 버퍼를 더 효율적으로 관리할 수 있도록 많은 연구가 이루어지고 있다.

기존의 디스크 버퍼 관리 정책으로 LRU(Least Recently Used), CF-LRU(Clean-First LRU), PT-LRU (Probabilistic Triplicate LRU), DPW-LRU(Dynamic Page Weight LRU), VBBMS(Virtual-Block-Based Buffer Management Scheme) 및 CLOCK 등을 예로 들 수 있다 [11-16]. LRU 정책은 가장 최근까지 참조된 적이 없는 페이지를 퇴거(Evict)시켜 더 많이 참조한 페이지들을 보존하게 된다. CF-LRU 정책은 가장 최근까지 참조된 적이 없으며 동시에 쓰기 연산이 이루어지지 않은 클린(Clean) 페이지를 우선적으로 퇴거한다. 그러나 LRU 정책 및 CF-LRU 정책 모두 미래의 재참조 가능성은 전혀 고려하지 않는다. 반면, CLOCK 정책의 경우 각 페이지마다 Second-Chance를 부여하여 미래의 참조 가능성을 고려한다. 그러나 쓰기 연산에 대한 가중치를 부여하지 않기 때문에 실제로 각 페이지마다 유효한 재참조 가능성을 부여하지는 않는다. PT-LRU 정책은 쓰기 연산 및 읽기 연산의 비율을 이용해 각각의 페이지를 3가지로 분할된 영역에 따로 저장하고, 이후 각 영역별 우선순위를 통해 페이지를 퇴거한다. DPW-LRU는 시간적 지역성, 퇴거 비용 및 페이지의 최신성을 고려하여 2가지로 분할된 영역에 저장한다. 그러나 PT-LRU와 DPW-LRU는 페이지를 유형에 따라 분류하므로 공간적 지역성은 고려될 수 없다. VBBMS 정책은 페이지에 대한 순차 요청 및 임의 요청을 구분하여 관리한다. 그러나 재참조 가능성을 고려하지는 않으며, 블록 기반 정책이 갖는 단편화 문제는 디스크 버퍼의 공간적 낭비를 초래한다.

앞서 언급한 것과 같이 LRU 정책을 기반으로 하는 여러 가지 디스크 버퍼 관리 정책들이 파생되어 왔으나 대부분의 정책들은 낸드 플래시 메모리의 특성을 고려하지 않는다. 낸드 플래시 메모리는 가급적 쓰기 연산을 발생시키지 않는 것이 태생적 수명 문제를 해결하는데 더 유리하다. 그러므로 현재 제안된 정책들은 개선될 여지가 있다.

III. 2WPR policy

1. 2WPR structure

본 논문에서는 메모리 버퍼를 이용하여 플래시 메모리의 쓰기 연산을 최소화할 수 있는 2WPR 알고리즘을 제안한다. 디스크 버퍼는 새로운 페이지 요청이 발생했을 때 각 페이지에 대한 참조 가능성을 계산하여 희생 페이지(Victim Page)를 선택한다. 만약 희생 페이지로 선택되면 해당 페이지는 바로 메모리로 퇴거되지 않고 두 번째 기회를 가진다. Fig. 1은 2WPR 정책의 구조를 나타내고 있다.

2WPR list는 새로운 페이지에 대한 읽기 혹은 쓰기 요청이 발생했을 때 이를 MRU 위치로 할당한다. 퇴거는 각 페이지의 참조 가능성 계산을 통해 이루어진다. 만약 재참조 가능성이 가장 낮은 페이지로 판명되면, 해당 페이지는 희생 페이지 리스트의 MRU 위치로 이동하게 된다.

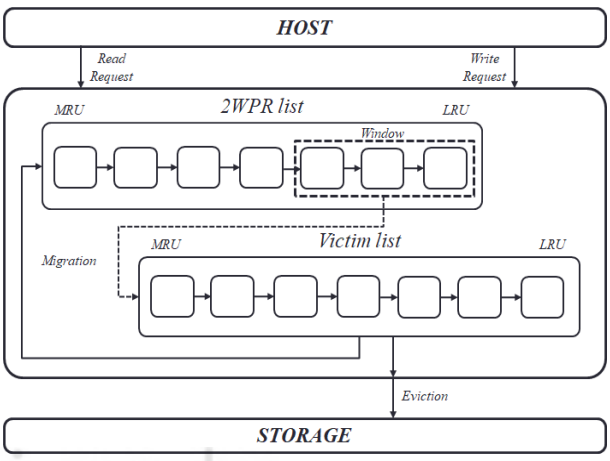


Fig. 1. 2WPR Structure

Victim list는 2WPR list에서 퇴거된 페이지들이 위치한다. 특정 페이지에 대한 새로운 요청이 발생했으나 해당 페이지가 이미 Victim list에 존재한다면 해당 페이지를 2WPR list의 MRU 위치로 이동시킨다. 희생 페이지 리스트에서의 퇴거는 클린 페이지를 우선으로 한다. 그러나 클린 페이지가 존재하지 않는다면 일반적인 LRU 정책과 마찬가지로 가장 마지막까지 참조되지 않았던 페이지를 선택하여 퇴거한다.

2. Page replacement in 2WPR

2WPR 정책에서 희생 페이지로 선택되는데 주요한 요소는 총 세 가지이다.

첫 번째는 시간적 지역성이다. 2WPR list에 저장되는 모든 페이지는 읽기 혹은 쓰기 횟수를 저장하며 둘을 합쳐 총 참조 횟수로 활용한다. 상대적으로 참조가 자주 되지 않은 페이지는 그렇지 않은 페이지에 비해 시간적 지역성이 더 낮다고 판단되어 희생 페이지로 선택될 가능성이 증가한다.

두 번째는 공간적 지역성이다. 연속되는 페이지가 서로 비슷한 참조 시간을 가질 때 해당 페이지들은 공간적인 지역성을 띄고 있음을 알 수 있다, 이를 이용하여 연속되는 페이지간 서로의 최종 참조 시간을 비교했을 때, 특정 페이지가 다른 페이지에 비해 공간적 지역성이 더 높다고 판단되면 희생 페이지로 선택될 가능성이 감소한다.

세 번째는 각 페이지에 대한 쓰기 연산의 가중치이다. 동일한 참조 횟수를 가지는 페이지라 할지라도 읽기와 쓰기 연산의 비율은 각각 다를 것이다. 이 때, 쓰기 연산에 가중치를 두어 쓰기 요청이 더 많았던 페이지가 그렇지 않은 페이지보다 2WPR list에 더 오래 머물게 한다.

참조 가능성은 총 네 가지 수식으로 계산되며 각 수식이 나타내는 것은 다음과 같다. 수식 1은 각 페이지의 총 참조 횟수를 계산하여 시간적 지역성을 판단한다. 수식 2는 공간적 지역성을 계산한다. 각 페이지는 인접한 페이지와의 최종 참조 시간 차이를 통하여 공간적 지역성을 판단한다. 수식 3은 쓰기 연산에 대한 가중치를 측정한다. 윈도우 내에서 이루어졌던 모든 참조 연산의 총합 중 각 페이지에서 발생한 쓰기 연산의 비율이 산출된다. 마지막 수식 4에서는 수식 1, 수식 2 그리고 수식 3에서 계산된 결과 값을 모두 합한다. 이렇게 도출되는 값이 더 작은 경우 희생 페이지로 선택되어 가능성이 큰 다른 페이지에 비해 Victim list로 이동할 가능성이 커진다.

수식 1: 특정 페이지의 총 참조 비율을 구한다. 이 때 리스트에서 발생한 모든 참조 횟수를 윈도우 크기만큼 나누고 분자로 정의한다. 각 페이지의 읽기 참조 횟수 및 쓰기 참조 횟수를 분모로 정의한다. 해당 값을 지수 함수로 계산하면 항상 0과 1 사이의 값이 도출되어 확률로 이용할 수 있다. 이 때 시간 지역성은 식 (1)과 같이 계산된다.

$$predict1(page_i) = e^{-1*(\frac{1+\sum_{j=1}^{w}(read_{page_j}+write_{page_j})*\frac{1}{w}}{(1+read_{page_i}+write_{page_i})})} \sim (0,1), 0 \leqq i \leqq w \quad (1)$$

수식 2: 특정 페이지에 대해 바로 앞 페이지와 참조되었던 시간의 차이를 공간적 지역성으로 정의한다. 이 때, 참조 시간은 age로 표기한다. 만약 각 페이지 간의 참조 시간 차이가 작다면 그렇지 않은 페이지에 비해 공간적 지역성이 큰 것으로 간주한다. 수식 1에서와 같이 이를 지수 함수로 계산하여 유효한 확률 값이 계산되도록 한다. 이 경우 공간 지역성은 식 (2)와 같이 도출된다.

$$predict2(page_{i-1}, page_i) = e^{-1*(1+abs(age_{page_{i-1}}-age_{page_i})*\frac{1}{2})} \sim (0,1), 0 \leqq i \leqq w \quad (2)$$

수식 3: 특정 페이지의 쓰기 연산 비율을 통해 앞으로의 쓰기 참조 확률을 계산한다. 윈도우 내에서 일어난 모든 쓰기 및 읽기 참조 횟수를 합한 값과 하나의 페이지에서 수행된 모든 쓰기 연산 횟수를 서로 나눈다. 결과 값을 지수 함수로 다시 계산하면 0과 1 사이의 값이 추출된다. 이 때, 쓰기 연산에 대한 가중치는 식 (3)과 같이 계산할 수 있다.

$$predict3(page_i) = e^{-1*(\frac{1+\sum_{j=0}^{w}(write_{page_j}+read_{page_j})*\frac{1}{w}}{1+write_{page_i}})} \sim (0,1), 1 \leqq i \leqq w \quad (3)$$

수식 4: 앞선 수식 1, 수식 2 , 그리고 수식 3을 모든 페이지마다 계산한다. 이후 각 수식마다의 결과 값을 모두 합하고 가장 작은 값을 갖는 페이지를 희생 페이지로 선택한다. 최종적으로 희생 페이지를 선택할 때 이용되는 수식은 식 (4)와 같다.

$$predict4 = (predict1(page_i) + predict2(page_{i-1}, page_i) + predict3(page_i)) \sim (0,3), 1 \leq i \leq w \quad (4)$$

요약하자면, 2WPR의 주된 전략은 다음과 같다.

(1). 2WPR list는 이후에도 참조될 가능성이 크며 그 중에서도 쓰기 연산이 자주 일어나는 페이지들이 저장된다. 또한 공간 지역성을 판별하여 연속되는 페이지들의 잦은 이주를 방지한다.

(2). Victim list는 2WPR list에서 소거되는 페이지들이 저장된다. 이는 각 페이지마다 참조될 수 있는 두 번째 기회를 부여하기 위함이다. 결과적으로 잦은 요청을 받는 페이지의 경우 디스크 버퍼에 더 오래 머물 수 있게 된다.

3. 2WPR scenario

Fig. 2와 Fig. 3은 2WPR 기법이 동작하는 전체적인 흐름을 보여준다. 임의의 페이지가 요청되면 2WPR list 및 Victim list 모두에 대해 해당 페이지가 존재하는지 확인한다. 만약 2WPR list에 히트했다면 해당 페이지는 2WPR list의 MRU 위치로 옮겨진다.

반면 버퍼의 Victim list에서 히트가 발생하면 해당 페이지를 2WPR 리스트의 MRU로 이주시킨다. 이 때, 각 리스트에 빈자리가 없다면 리스트간 페이지 교환이 일어날 수 있다. Fig. 2은 이러한 리스트 사이의 페이지 교환을 나타내고 있다.

Fig. 3은 페이지 폴트가 발생했을 때를 보여준다. 이 때, 각 리스트는 희생 페이지를 선택한다. 2WPR list의 희생 페이지는 Victim list로 이동하며, Victim list의 희생 페이지는 플래시 메모리로 퇴거된다.

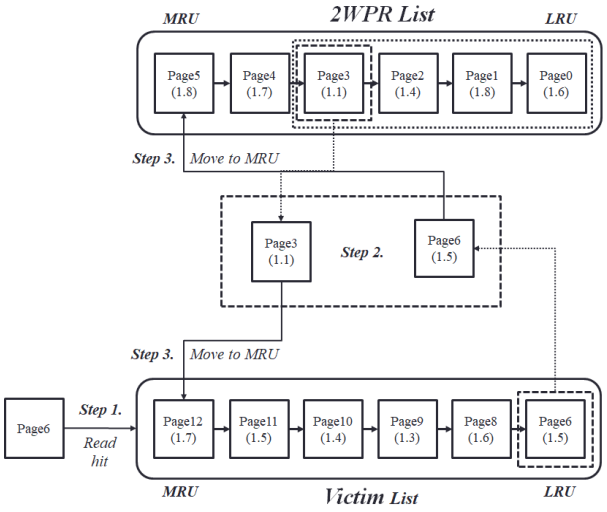


Fig. 2. Page Hit in Victim List

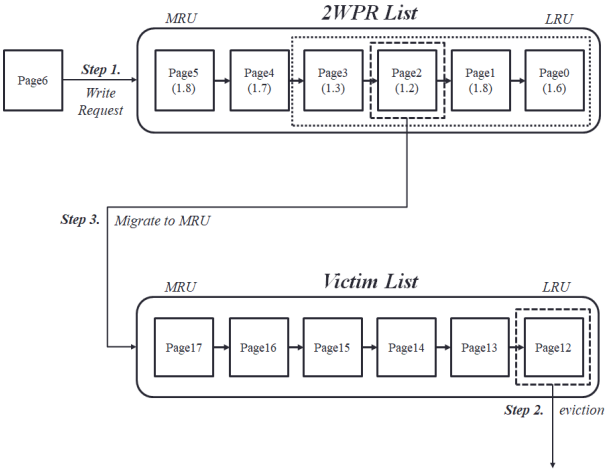


Fig. 3. Page Fault Procedure

4. 2WPR algorithm

Algorithm 1은 2WPR 정책의 동작 상세를 보여준다. 새로운 페이지가 요청되었을 때, 2WPR list에서 히트하였다면 해당 페이지는 2WPR list의 MRU 위치로 이동한다 (1-2행). 그러나 페이지 폴트가 발생했고 2WPR list에 빈 자리가 없다면 희생 페이지를 선정해 Victim list로 이주시킨다 (10-11행, 16행). 이 때, Victim list에 빈 공간이 없다면 희생 페이지를 선택해 플래시 메모리로 퇴거 시킨다 (12-15행). 만약 Victim list에서 히트 하였다면 해당 페이지는 2WPR list로 이동한다 (3행, 8행). 그러나 2WPR list에 빈 공간이 없다면 희생 페이지를 선정해 Victim list로 이주 시킨다 (4-7행).

Algorithm 1. 2WPR Algorithm

input: L_{2WPR} : 2WPR list
L_{victim} : Victim list

```
1.  if page is in L_2WPR then
2.    Insert page to MRU of L_2WPR;
3.  else if page is in L_victim then
4.    if there is no free space in L_2WPR
5.      victim = SelectVictim(L_2WPR);
6.      Move victim to MRU of L_victim;
7.    end if
8.    Insert page to MRU of L_2WPR;
9.  else
10.   if there is no free space in L_2WPR
11.     victim = SelectVictim(L_2WPR);
12.     if there is no free space in L_victim
13.       victim2 = SelectVictim(L_victim);
14.       Write victim2 to flash memory;
15.     end if
16.     Move victim to MRU of L_victim;
17.   end if
18.   Insert page to MRU of L_2WPR;
19. end if
```

Algorithm 2. SelectVictim Function

```
input:  L_2WPR : 2WPR list
        L_victim : Victim list
output: the index of a victim page for replacement
1.  If L is in L_victim
2.    for sit = LRU to MRU
3.      Select sit page in L_victim as q;
4.      if q is clean
5.        victim = q;
6.        break;
7.      end if
8.    end for
9.    if victim is null
10.     Select LRU page in L_victim as victim;
11.   end if
12. else
13.   Select min probablity in L_2WPR as victim;
14. end if
15. return the index of a victim;
```

Algorithm 2는 희생 페이지를 선택하는 SelectVictim 함수를 나타낸다. 만약 Victim list에서 희생 페이지를 선택한다면 클린 페이지를 우선적으로 선택하고 클린 페이지가 존재하지 않는다면 Victim list의 LRU에 위치한 페이지를 선택한다 (1-11 행). 반대로 2WPR list에서 희생 페이지를 선택한다면 앞서 제시된 수식들을 이용하여 참조 가능성을 계산한 뒤, 재참조 가능성이 가장 낮은 페이지를 선택한다 (12-14행).

IV. Performance evaluation

1. Experimental environment

알고리즘의 성능을 평가하기 위한 실험은 워크로드 기반 이벤트 구동형 시뮬레이터(Event-driven Simulator)를 구현하여 수행되었다. 실험을 위한 시스템은 64GB의 플래시 메모리에 4MB에서 64MB까지의 버퍼를 가지도록 구성되었다. 이 때, 하나의 페이지는 4KB로 구성된다.

실험 비교 대상은 기본적인 관리 정책인 LRU와 플래시 메모리에 대한 쓰기 연산을 감소시킨 정책인 CF-LRU로 선정하였다. 이 때, 실험 대상인 플래시 메모리는 Table 3의 특성을 따른다 [4-10].

Table 4는 각 워크로드의 상세를 나타내고 있다. 워크로드는 일반적으로 가장 많이 이용되고 있고 보편적인 읽기와 쓰기 연산의 형태를 담고 있는 것들로 선택하였다.

Table 3. Simulation Parameter

Attribute	Value
Block size	64 pages
Page size	4 KB
Write latency	200 μs / page
Erase latency	1.2 ms / page
Read latency	21 μs / page
Durablity	100,000

Table 4. Workloads for Simulation

Type	Trace	Total operations	Read/Write ratio
T_1	ascii	1,016,678	79%/21%
T_2	Financial1	36,115,179	85%/15%
T_3	Financial2	17,693,541	22%/78%

2. Results and Discussion

Table 5와 Table 6은 각 정책의 플래시 메모리에 대한 히트율과 쓰기 횟수 그리고 모든 연산 할당 및 총 지연 시간을 보여준다. 해당 값들을 통해 2WPR 정책이 플래시 메모리에 가해지는 부담을 줄일 수 있음을 알 수 있다. 그러나 디스크 버퍼의 크기가 커질수록 플래시 메모리에 대한 연산 할당 및 지연 시간이 점진적으로 평준화되는 모습을 볼 수 있다. 이를 통해 디스크 버퍼의 크기를 늘리기만 하는 것은 오히려 버퍼의 낭비를 초래할 가능성이 있다는 점을 확인할 수 있다.

Fig. 4는 버퍼의 크기에 따른 각 워크로드별 히트율을 비교한 것이다. 실험 결과를 통해 2WPR 정책이 다른 두 정책보다 더 높은 히트율을 보장한다는 점을 알 수 있다. 제안된 기법은 LRU 대비 평균 4%, CF-LRU 대비 평균 3% 향상된 히트율을 보여준다. 이는 2WPR 정책이 플래시 메모리에 가해지는 부담을 줄일 뿐 아니라 히트율 역시 보장하는 정책임을 증명한다. 이를 통해 2WPR 정책이 각 페이지의 지역성을 이용해 재참조 가능성을 고려하고 페이지를 더욱 효율적으로 디스크 버퍼에 잔류시킬 수 있다는 것을 확인 할 수 있다.

Fig. 5는 각 워크로드별 플래시 메모리에 할당되는 쓰기 연산의 횟수를 각 버퍼의 크기별로 비교한 것이다. 제안된 2WPR 정책이 LRU와 CF-LRU 정책보다 각각 평균 2%, 3% 더 적은 쓰기 연산을 플래시 메모리에 할당했음을 알 수 있다. 이러한 결과는 2WPR 정책에서의 쓰기 연산 가중치 부여가 유의미하게 동작하고 있음을 증명한다.

Fig. 6은 버퍼 크기에 따른 각 워크로드별 플래시 메모리에 할당된 총 읽기/쓰기 연산 횟수를 비교한 것이다. 해당 실험 결과를 통해 버퍼의 크기가 클 때는 읽기 연산이 다른 정책만큼 많이 발생할 수 있다는 점을 알 수 있는데,

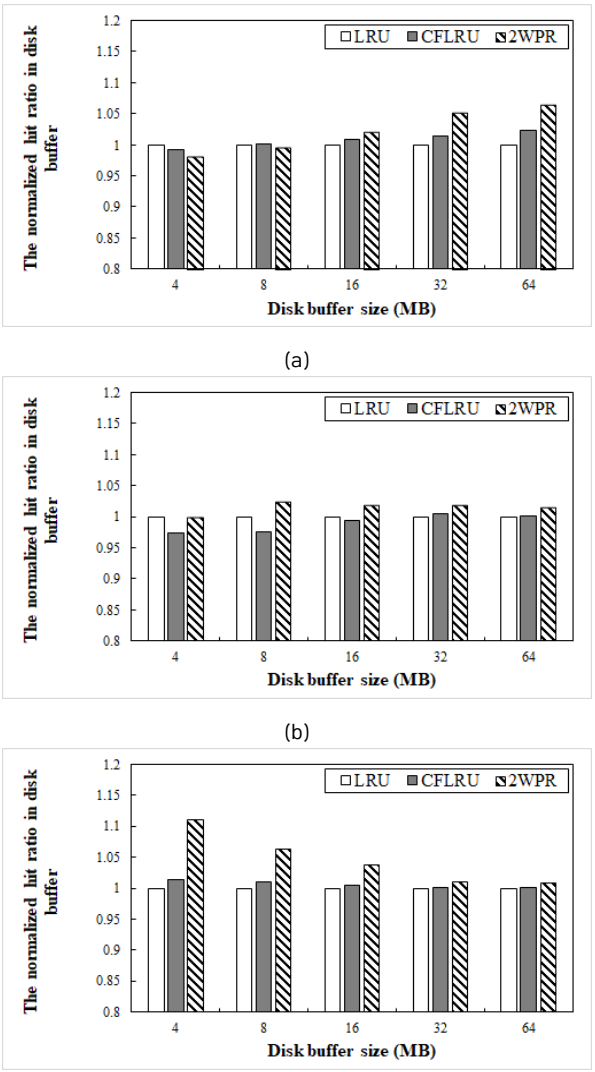


Fig. 4. Hit Ratio (a) T_1, (b) T_2, (c) T_3

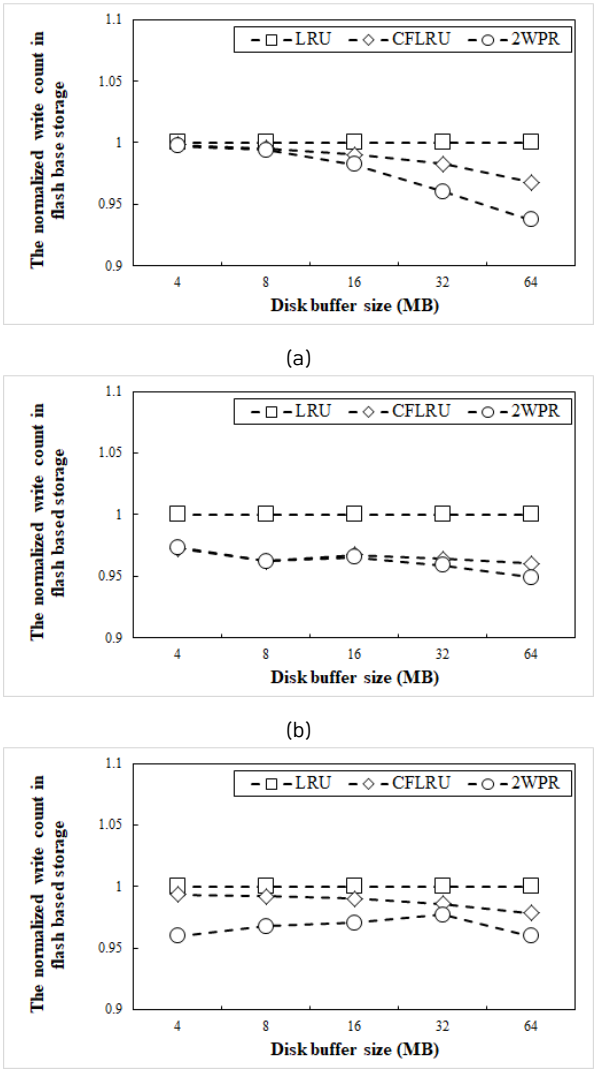


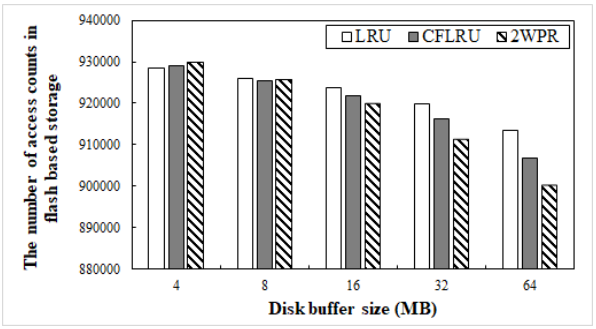
Fig. 5. Normalized Write Counts (a) T_1, (b) T_2, (c) T_3

Table 5. Normalized hit ratio and write count

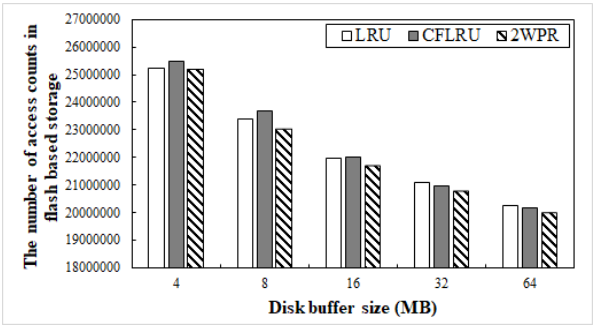
Buffer Size(MB)	Normalized hit ratio					Normalized write count				
	4	8	16	32	64	4	8	16	32	64
LRU	1	1	1	1	1	1	1	1	1	1
CF-LRU	1.0132	1.0105	1.0047	1.0011	1.0006	0.9938	0.9921	0.9902	0.9863	0.9788
2WPR	1.1094	1.0615	1.0369	1.0097	1.0069	0.96	0.9681	0.9705	0.9775	0.9598

이는 쓰기 가중치 적용 정책에 의해 디스크 버퍼가 잘 관리되고 있다는 반증이 된다. 특히, 쓰기 연산이 많은 T_3에 대비 LRU는 평균 2%, CF-LRU는 평균 3% 더 적은 연산을 플래시 메모리에 할당하였다.

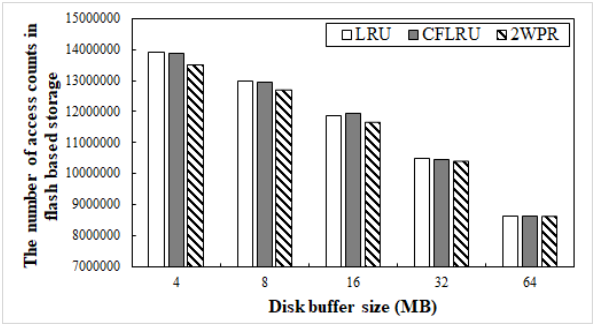
Fig. 7은 각 워크로드별로 실행하는데 소요된 지연 시간을 나타낸다. 해당 결과는 플래시 메모리에 할당되는 쓰기 연산이 줄어든 것과 비슷한 양상을 보임을 알 수 있다. 이는 플래시 메모리의 연산 속도가 각 연산에 대해 비대칭적이고 특히 쓰기 연산의 비중이 크기 때문이다. 버퍼의 크기가 커질수록 더 적은 지연 시간을 필요로 하는 경향을 보이며, 결과적으로 LRU 대비 평균 3%, CF-LRU 대비 평균 2% 의 향상을 이끌어냈다.



(a)

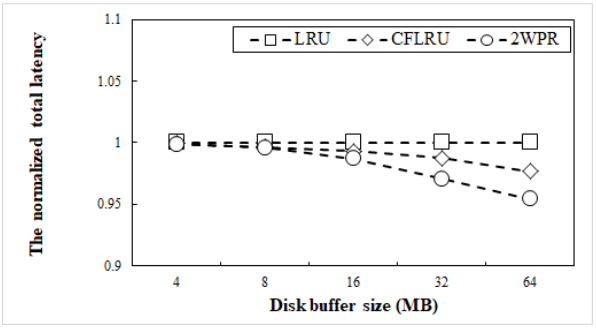


(b)

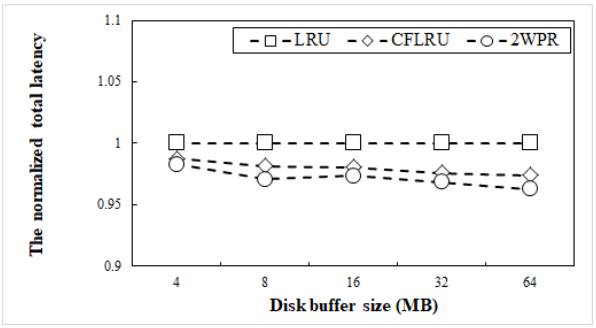


(c)

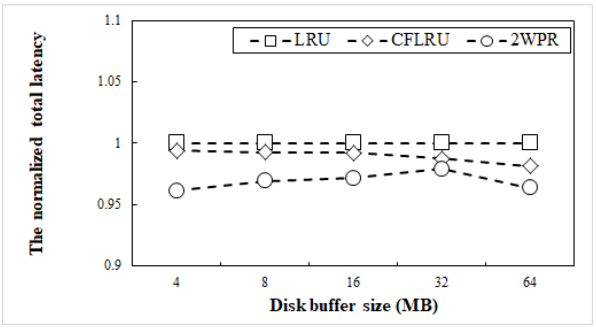
Fig. 6. Read/Write Access Counts (a) T_1, (b) T_2, (c) T_3



(a)



(b)



(c)

Fig. 7. Execution Latency (a) T_1, (b) T_2, (c) T_3

Table 6. Total Access Counts and Normalized Latency

Buffer Size(MB)	Total access counts(10^3)					Normalized total latency				
	4	8	16	32	64	4	8	16	32	64
LRU	13927	12989	11867	10479	8656	1	1	1	1	1
CF-LRU	13871	12931	11932	10461	8640	0.9940	0.9925	0.9922	0.9878	0.9813
2WPR	13500	12686	11636	10394	8578	0.9612	0.9692	0.9718	0.9793	0.9638

V. Conclusions

본 논문에서는 디스크 버퍼를 더 효율적으로 관리하기 위한 2WPR 정책에 대해 소개하였다. 2WPR 정책은 각 페이지별 재참조 가능성 및 쓰기 연산에 대한 가중치를 이용하여 디스크 버퍼에 더 오래 잔류시킬 페이지를 결정한다. 또한 2WPR 정책은 디스크 버퍼를 2WPR list 및 Victim list로 나누어 관리하며, 각 페이지는 제안된 알고리즘에 의해 두 개의 list 사이에서 이동하거나 Victim list에서 플래시 메모리로 퇴거될 수 있다. 제안된 정책은 다양한 워크로드에서 디스크 버퍼에 대한 히트율이 LRU 대비 평균 4% 상승하는 것을 확인하였으며, 동시에 플래시 메모리에

할당되는 쓰기 연산 및 지연 시간 역시 LRU 대비 각각 평균 2%와 3%만큼 감소한 것을 확인하였다.

제안된 2WPR 정책은 가능성을 판별하는 영역의 크기가 정적으로 고정되어 있다. 이는 퇴거 가능성을 도출하는 과정에서 불필요한 연산을 수행할 가능성이 있다. 따라서 향후 연구로 해당 영역의 크기를 동적으로 관리하는 연구를 진행할 예정이다.

ACKNOWLEDGEMENT

This research was supported by Basic Science Research Program through the National Research Foundation of Korea(NRF) funded by the Ministry of Education (2017R1D1A1A09000654).

REFERENCES

[1] R. Caceres, F. Douglis, Kai Li and B. Marsh, “Operating system implications of solid-state mobile computers”, Proceedings of IEEE 4th Workshop on Workstation Operating Systems. WWOS-III, pp. 21-27, October 1993.

[2] Lawton, George, “Improved flash memory grows in popularity”, Computer, Vol. 39, No. 1, pp. 16-18, 2006.

[3] Kim, Han-joon, and Sang-goo Lee, “A new flash memory management for flash storage system”, Proceedings. Twenty-Third Annual International Computer Software and Applications Conference (Cat. No. 99CB37032), pp 248-289, 1999.

[4] Dong, Xiangyu and Xu, Cong and Xie, Yuan and Jouppi, Norman P, “Nvsim: A circuit-level performance, energy, and area model for emerging nonvolatile memory”, IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, Vol. 31, No. 7, pp. 994-1007, 2012.

[5] Saxena, Mohit and Swift, Michael M, “FlashVM: Virtual Memory Management on Flash.”, USENIX Annual Technical Conference, 2010.

[6] Xie, Yuan, “Emerging Memory Technologies: Design, Architecture, and Applications”, pp. 15-50, 2013.

[7] Yang, Ming-Chang and Chang, Yuan-Hao and Tsao, Che-Wei and Huang, Po-Chun, “New ERA: New efficient reliability-aware wear leveling for endurance enhancement of flash storage devices”, 2013 50th ACM/EDAC/IEEE Design Automation Conference (DAC), pp. 1-6, 2013.

[8] Hong, Seongcheol and Shin, Dongkun, “NAND flash-based disk cache using SLC/MLC combined flash memory”, 2010 International Workshop on Storage Network Architecture and Parallel I/Os, pp. 21-30, 2010.

[9] Karmakar, Supriya, “Quantum dot gate non-volatile memory as single level cell (SLC), multi-level cell (MLC) and triple level cell (TLC)”, 2018 IEEE Long Island Systems, Applications and Technology Conference (LISAT), pp. 1-5, 2018.

[10] Bennett, Sorcha and Sullivan, Joe, “The Characterisation of TLC NAND Flash Memory, Leading to a Definable Endurance/Retention Trade-Off”, International Journal of Computer, Electrical, Automation, Control and Information Engineering, Vol. 10, pp.716-723, 2016.

[11] Panagakis, Antonis and Vaios, Athanasios and Stavrakakis, Ioannis, “Approximate analysis of LRU in the case of short term correlations”, Computer Networks, Vol. 52, No. 6, pp. 1142-1152, 2008.

[12] Park, Seon-yeong and Jung, Dawoon and Kang, Jeong-uk and Kim, Jin-soo and Lee, Joonwon, “CFLRU: a replacement algorithm for flash memory”, Proceedings of the 2006 international conference on Compilers, architecture and synthesis for embedded systems, pp. 234-241, 2006.

[13] Nicola, Victor F and Dan, Asit and Dias, Daniel M, “Analysis of the generalized clock buffer replacement scheme for database transaction processing”, Vol. 20, No. 1, pp. 35-46, 1992.

[14] Cui, Jinhua and Wu, Weiguo and Wang, Yinfeng and Duan, Zhangfeng, “PT-LRU: a probabilistic page replacement algorithm for NAND flash-based consumer electronics”, IEEE Transactions on Consumer Electronics, Vol. 60, No. 4, pp. 614-622, 2014.

[15] Yuan, Youwei and Zhang, Jintao and Han, Guangjie and Jia, Gangyong and Yan, Lamei and Li, Wanqing, “DPW-LRU: An Efficient Buffer Management Policy Based on Dynamic Page Weight for Flash Memory in Cyber-Physical Systems”, IEEE Access, Vol. 7, pp. 58810-58821, 2019.

[16] Du, Chenjie and Yao, Yingbiao and Zhou, Jie and Xu, Xiaorong, “VBBMS: A Novel Buffer Management Strategy for NAND Flash Storage Devices”, IEEE Transactions on Consumer Electronics, Vol. 65, No. 2, pp. 134-141, 2019.

Authors

Won Ho Lee received a B.S degree in Department of Computer Engineering from Yeungnam University, Korea in 2014. He is currently a M.S. student in Department of Computer Engineering at Yeungnam University. His current research interests include advanced processor architecture, low-power mobile embedded systems and non-volatile memory.

Jong Wook Kwak received a B.S. degree in Computer Engineering from Kyungpook National University, Daegu, Korea in 1998, a M.S. degree in Computer Engineering from Seoul National University, Seoul, Korea in 2001, and a Ph.D. degree in Electrical Engineering and Computer Science from Seoul National University, Seoul, Korea in 2006. From 2006 to 2007, he worked as a Senior Engineer in the SoC R&D Center, at Samsung Electronics Co., Ltd. During 2011~2012, he was a Guest Researcher at the Research Institute of Advanced Computer Technology, Seoul National University. During 2012~2013, he was a Visiting Scholar at the Georgia Institute of Technology, Atlanta, GA, USA. As a Head Director, he led DREAM Software Human Resource Training Center from 2014~2015. During 2018~2019, he was a Visiting Scholar at Arizona State University, Tempe, AZ, USA. He is currently a professor in the Department of Computer Engineering, Yeungnam University. His research interests include advanced processor architecture, low-power mobile embedded systems, and high performance parallel computing.