논문 2018-13-18

WAP-LRU : 플래시 스토리지 시스템에서 쓰기 패턴 분석 기반의 하이브리드 디스크 버퍼 관리 기법

(WAP-LRU: Write Pattern Analysis Based Hybrid Disk Buffer Management in Flash Storage Systems)

김 경 민, 최 준 형, 곽 종 욱*
(Kyung Min Kim, Jun-Hyeong Choi, Jong Wook Kwak)

Abstract : NAND flash memories have the advantages of fast access speed, high density and low power consumption, thus they have increasing demand in embedded system and mobile environment. Despite the low power and fast speed gains of NAND flash memory, DRAM disk buffers were used because of the performance load and limited durability of NAND flash cell. However, DRAM disk buffers are not suitable for limited energy environments due to their high static energy consumption. In this paper, we propose WAP-LRU (Write pattern Analysis based Placement by LRU) hybrid disk buffer management policy. Our policy designates the buffer location in the hybrid memory by analyzing write pattern of the workloads to check the continuity of the page operations. In our simulation, WAP-LRU increased the lifetime of NAND flash memory by reducing the number of garbage collections by 63.1% on average. In addition, energy consumption is reduced by an average of 53.4% compared to DRAM disk buffers.

Keywords : Flash memory, Nonvolatile memory, Hybrid disk buffer, Buffer replacement policy, LRU

Ⅰ. 서 론

최근 스마트 폰, 스마트 패드를 비롯해 IoT 등 소형 전자기기 및 임베디드 시스템의 고성능 프로세싱의 수요 증가로 인해 스토리지로 NAND 플래시 메모리 (이하 플래시 메모리) 기반의 저장장치 사용이 급증하고 있다. 플래시 메모리는 높은 집적도와 내구성, 빠른 접근 시간과 저전력이라는 특징을 가지며, 이로 인해 기존의 디스크 기반 스토리지를 대체하거나 모바일 기기의 스토리지로 사용된다.

반면, 플래시 메모리는 쓰기 연산에서 잔여 공간이 부족하면 기존 할당 영역을 덮어쓰지 못해 쓰기 이전에 블록 단위로 삭제 연산을 수행해야 한다. 삭제 연산은 지연 시간과 에너지 소모의 부하가 크고 한 셀에 대한 삭제 연산의 횟수가 제한되어 있어 잦은 삭제 연산은 플래시 메모리의 성능 하락과 수명감소를 유발한다 [1].

이를 해결하기 위해 DRAM으로 구성된 디스크 버퍼를 이용하여 스토리지의 성능 향상을 이루는 기법들이 연구되었다. 버퍼 관리 정책으로 LRU, CLOCK 등 전통적인 알고리즘과 이를 개선한 여러 DRAM 디스크 버퍼 기반의 다양한 기법들이 제안되었다 [2-3]. 하지만, DRAM의 재충전 (Refresh)으로 인한 대기 에너지 소모가 전체 시스템에서 차지하는 비율이 40% 이상으로 높고 기술 발전으로 NVRAM의 집적도가 상승하게 됨으로써 한정된 에너지를 기반으로 하는 IoT 기기나 모바일 응용에서는 버퍼를 DRAM과 NVRAM의 하이브리드 구조로 사용하는 연구가 활발하게 진행되고 있다 [4-6].

본 논문에서는 버퍼 메모리의 특성에 따라 읽기/쓰기 페이지를 분리하여 배치하는 하이브리드 디스크 버퍼 관리기법을 제안한다. 요청되는 페이지가

*Corresponding Author (kwak@yu.ac.kr)
Received: May 8 2018, Revised: May 24 2018, Accepted: June 1 2018
K.M. Kim, J.H. Choi, J.W. Kwak: Yeungnam University
※ 본 논문은 2017년도 정부(교육부)의 재원으로 한국연구재단의 지원을 받아 수행된 기초연구사업임 (2017R1D1A1A09000654).

 2018 June 13(3) 151-160
ISSN : 1975-5066
http://dx.doi.org/10.14372/IEMEK.2018.13.3.151

순차 접근 경향 혹은 임의 접근 경향이 있는지를 판단하여 버퍼 영역의 메모리 특성에 맞게 페이지를 나누어 배치하고 쓰기 적중률을 상승시켜 가비지 컬렉션의 횟수를 감소시킨다. 결과적으로 삭제 및 쓰기 연산을 감소시켜 플래시 메모리의 수명을 증가시키고 에너지 소모를 개선한다.

본 논문의 구성은 다음과 같다. 2장에서는 메모리 소자의 특성과 버퍼 메모리의 관리 정책에 대한 관련 연구를 설명한다. 3장에서는 제안된 SSD 시스템의 전체 구조와 제안된 알고리즘에 관해 설명한다. 4장에서는 실험의 결과와 평가를 서술하며 5장에서 결론을 맺는다.

II. 관련 연구

본 장에서는 최근의 메모리 연구 현황에 관해 설명하고, FTL과 디스크 버퍼를 관리 하는 기존의 정책들을 소개한다.

1. 메모리 기술 현황

NVRAM은 플래시 메모리를 제외하고는 DRAM과 비교해 낮은 집적도를 가지고 있어 메인 메모리나 버퍼보다는 L3 캐시와 같은 CPU 하위의 캐시로 주로 사용되었다 [7]. 그러나 최근의 기술 발전으로 인해 NVRAM의 집적도가 개선된 반면 DRAM의 에너지 소모 및 집적도는 한계에 도달하여 NVRAM과 DRAM을 버퍼로 함께 구성하는 사례가 늘고 있다. 근래 메모리 기술의 추이를 아래의 표 1에 요약하였다 [8-12].

표 1의 에너지는 셀 당 에너지 소모 비율을 의미하며 수식 (1)을 이용하여 계산하였다. 밀도 F값은 공정에서 프로그래밍 펄스에 따라 달라지는 값으로 같은 공정에서의 상대적인 비율을 의미한다.

$$E_{write} = I_{reset} * V_{dd} * t_{write} \quad (1)$$

PCM은 쓰기 속도가 느리고 동적 에너지 소모 비중이 크지만, 단위 칩당 집적도가 높은 특성으로 인해 하이브리드 디스크 버퍼 메모리의 NVRAM으로 사용되어왔다 [9]. 다만 PCM은 수명이 제한되어 있고, 쓰기 에너지와 지연 시간이 커서 저전력, 모바일 등의 환경에는 적합하지 않다.

STT-MRAM은 에너지 및 속도 측면에서 DRAM과 유사한 성질을 가지면서도 재충전 전력을 필요로 하지 않지만, 집적도가 낮은 문제가 있다. 하지만 저전력 기반의 환경에서 버퍼를 구성할 때

표 1. 메모리 기술 지표
Table 1. Memory technology metrics

	DRAM	PCM	STT MRAM	NAND Flash
Energy	<1 pJ	18 pJ	1 pJ	100 pJ
Read Latency	5 ns	80 ns	10 ns	30 us
Write Latency	50 ns	150 ns	5 ns	100 μs
Durability	-	10^8	-	10^6
Density	7 F	4 F	12 F	<4 F

작은 용량을 사용한 STT-RAM을 이용하여 하이브리드 디스크 버퍼를 구성하면 PCM과 DRAM이 가지고 있는 정적/동적 전력, 쓰기 속도 측면의 한계를 극복할 수 있을 것으로 기대된다 [9-12].

2. 플래시 메모리의 FTL 정책

FTL (Flash Transition Layer)은 플래시 메모리가 기존의 파일 시스템과의 호환성을 위하여 추가적으로 설계된 계층이다. FTL의 주된 역할은 논리적 블록 매핑 기능과 플래시 메모리의 공간을 확보하는 가비지 컬렉션이 있다.

논리적 블록 매핑은 매핑 단위에 따라 페이지 레벨, 블록 레벨, 로그 블록 등으로 구분 할 수 있다. 페이지 레벨 매핑은 테이블의 한 영역을 페이지 단위로 관리함으로써 데이터를 유연하게 처리할 수 있는 특징이 있다. 블록 레벨 매핑은 데이터를 블록 단위로 관리하여 관리하며, 오프셋을 이용하여 특정 페이지에 접근한다. 그로 인하여 임베디드 시스템과 같이 하드웨어의 공간적 제약이 있는 환경에서 주로 사용된다. 로그 블록 매핑은 호스트에서 요청하는 쓰기 명령을 순차적으로 로그 블록에 기록하고 로그 블록이 가득 차면 같은 논리적 블록 번호를 가지는 블록의 데이터와 병합하여 새로운 블록으로 기록하는 방법이다.

가비지 컬렉션은 플래시 메모리 내의 공간이 부족한 경우에 새로운 블록을 확보하기 위한 기법이다. 가비지 컬렉션 수행 시, 유효 페이지 이주와 블록 삭제를 해야 하므로 플래시 메모리의 성능 저하가 발생한다. 가비지 컬렉션으로 인한 성능 저하를 극복하기 위하여 다양한 형태의 연구가 진행되었으며, 그중에서 GA (Greedy Algorithm)와 CB (Cost Benefit)가 대표적이다.

GA는 블록 내에 남아있는 유효한 페이지가 가

장 적은 블록을 희생 블록으로 선정하여 페이지의 이주를 최소화하는 특징이 있다. 이와 달리 CB는 블록 내의 페이지와 페이지들의 경과 시간도 고려해서 희생 블록을 선정하므로 장기적으로 유효 페이지의 이주를 감소시킬 수 있다 [13].

3. 하이브리드 기반 버퍼 교체 정책

전통적으로 사용된 페이지 기반 버퍼 교체 정책은 LRU (Least Recently Used)와 CLOCK이 사용되었다. LRU 알고리즘은 가장 오랫동안 사용되지 않은 페이지를 교체하는 방식으로 동작하고 CLOCK 알고리즘은 참조된 페이지에 대한 Second Chance를 부여하는 방식으로 참조된 페이지를 오랫동안 버퍼에 유지한다. LRU와 CLOCK은 페이지가 시간 지역성을 가지고 있다는 전제하에 동작하므로 페이지 참조가 국소적인 범위에서 발생하는 환경에서는 우수한 성능을 보이지만, 하이브리드 메모리 구성에서는 메모리의 특성을 반영하지 못하며 페이지 참조가 혼재된 상황에서는 좋지 않은 성능을 보일 수 있다.

한편, 하이브리드 버퍼 관리 정책은 버퍼를 DRAM과 NVRAM으로 함께 구성하며, 이로 인해 버퍼를 관리할 때 페이지 단위뿐만이 아니라 블록 단위로 버퍼 교체 정책을 혼합하여 사용하는 방식이다. 하이브리드 메모리 구성에서 DRAM을 페이지 단위로, 부가되는 NVRAM을 블록 단위로 관리하며 최근 연구된 기법으로는 CBM과 CLOCK-DNV 등이 있다 [5, 6].

CBM은 작은 크기의 쓰기 버퍼로 NVRAM을 사용하고 읽기 버퍼로 DRAM을 사용하여 버퍼를 구성한다. 해당 기법은 NVRAM에서 블록이 플래시 메모리로 쓰일 때 DRAM에서 같은 논리 블록 번호를 갖는 페이지들을 병합하여 순차 쓰기 비율을 높인다. 그러나 이 기법은 소량의 NVRAM을 쓰기 버퍼로 사용하기 때문에 쓰기 연산의 빈도가 많은 환경에서는 낮은 적중률을 보이는 문제점이 있다 [5].

CLOCK-DNV 알고리즘은 DRAM과 NVRAM을 CLOCK 알고리즘으로 관리하면서 읽기/쓰기에 대한 요청을 관리한다. DRAM에서의 여유 공간이 부족할 때 쓰기 연산이 발생한 페이지를 우선하여 희생 페이지로 선정하고 이를 NVRAM으로 이주시킨다. 쓰기 참조가 없는 페이지는 DRAM에서 바로 제거하여 쓰기 페이지에 대한 참조율을 높인다. 그러나 주어진 방식은 NVRAM을 쓰기 버퍼로만 활용하기 때문에 읽기/쓰기 연산이 혼합된 환경에서는 읽기 연산에 대한 버퍼 동작의 비효율성이 야기된다 [6].

Ⅲ. 본 론

본 논문에서는 하이브리드 메모리 구성으로 에너지 소모를 감소시키면서 가비지 컬렉션으로 인한 플래시 메모리의 수명감소를 최소화하는 하이브리드 디스크 버퍼 관리기법을 제안한다. 본 논문에서 제안하는 하이브리드 디스크 버퍼 관리기법은 LRU를 기반으로 하여 DRAM 버퍼 영역은 페이지 단위로 NVRAM 버퍼 영역은 블록 단위로 관리한다. NVRAM 버퍼 영역은 논리적 블록으로 구성하여 플래시 메모리로의 쓰기가 발생할 때 순차 쓰기를 수행한다. 페이지를 배치할 때 워크로드의 선행 분석(Prior profiling)을 통해 페이지의 연속된 정도에 따라 이를 나누어 배치함으로써 읽기/쓰기 요청이 분산된 작업에서 효율적인 버퍼 활용이 가능하다. 제안된 기법은 기존 버퍼 알고리즘들과 비교하여 낮은 에너지 소모와 적은 가비지 컬렉션으로 플래시 메모리의 수명 증가를 기대할 수 있다.

1. 제안 동기

실시간으로 데이터가 처리되는 워크로드들은 읽기/쓰기 요청이 동시에 발생하며 쓰기 요청은 전체 동작에서 적은 비율을 차지하면서도 불균형하게 분포되어 있다. 불규칙한 쓰기 요청의 분포로 인해 버퍼의 적중률은 낮아지고 가비지 컬렉션 시 많은 오버헤드가 발생하여 플래시 메모리의 성능을 감소시키고 추가적인 에너지 손실을 일으킨다. 따라서 제안하는 버퍼 관리기법은 구간에서 발생하는 요청의 연속적인 정도를 기록한다. 기록된 정보를 참조해서 하이브리드 메모리에서 페이지가 배치될 위치를 결정하여 플래시 메모리에 대한 직접 접근을 감소시킨다.

제안하는 알고리즘 기법은 호스트 요청의 쓰기 비율에 따른 버퍼 매니저의 상태 변화를 활용하여 DRAM 혹은 NVRAM 버퍼 영역에 대해 요청을 나누어 배치하는 전략을 핵심으로 한다. 따라서 배치 전략 수립을 위해 애플리케이션의 특성을 반영하는 워크로드의 선행 분석이 필요하다. 워크로드 선행 분석 과정은 호스트 인터페이스의 요청을 임의의 단위로 표본을 뽑아 샘플링된 구간에서 평균적으로 나타나는 연속된 쓰기 요청의 페이지 크기와 단일 쓰기 요청의 평균 및 최대 크기를 수집한다. 수집된 연속 쓰기 요청의 페이지 크기는 현재 호스트의 요청을 균형된 상태 (Balanced)인지 쓰기가 집중된 상태 (Write-Intensive)인지 버퍼 매니저가 판단하

표 2. 2OS 테이블 변수 목록
Table 2. 2OS table variable list

I_{length}	The maximum number of host request in the interval
$R(n)_{op}$	The operation type of host request (Read/Write)
$R(n)_{size}$	The size of host request
W_{eval}	The number of continuous write page for state change
F_{writes}	The frequency of non-contiguous write request in the interval
P_{state}	The deployment policy state (BAL/WI)
T_{state}	The minimum continuous write value for policy change
T_{read}	The maximum number of read request for DRAM placement
T_{write}	The maximum number of write request for DRAM placement

는 기준으로 사용한다. 단일 쓰기 요청의 평균 및 최대 페이지 크기는 호스트 요청이 임의 접근에 대한 요청인지 순차적인 특성을 가지는 요청인지에 대해 구분하여 DRAM과 NVRAM 버퍼 영역에 나누어 배치하는 기준으로 사용한다. 실험을 통해 워크로드에서 나타나는 연속된 페이지 쓰기 요청은 전체 빈도의 상위 25% 구간을 배치 정책을 위한 최소 임계치로 설정하며 단일 페이지의 값 중 중앙값과 상위 25% 값의 평균을 순차성 판단의 기준값으로 사용하였고, 이때 가비지 컬렉션의 횟수가 감소하는 것을 확인하였다. 분석된 워크로드의 결과는 데이터 추출에 사용된 시스템과 유사한 범주에 속하는 시스템에 대해 제안 기법을 적용할 때 최적화된 권고 수치로 제공할 수 있다.

2. WAP-LRU 알고리즘

본 논문에서 제안하는 쓰기 패턴 분석 기반의 LRU 배치 정책 기법 (WAP-LRU, Write pattern Analysis based Placement by LRU)은 실행 전 워크로드 분석을 통해 읽기/쓰기 요청의 특성을 파악하고 이를 기반으로 DRAM 버퍼 영역 혹은 NVRAM 버퍼 영역에 각각의 페이지를 구분하여 배치하여 LRU 기법을 사용하여 페이지 출입을 관리한다. 쓰기 요청의 페이지 크기를 실시간으로 수집하여 2OS (Operation Occurrence Status)에 기록한다. 표 2와 같이, 2OS는 상태 변화와 배치 위치를 결정하기 위한 상수값과 현재 구간에서의 평균 쓰기 페이지를 평가하기 위한 변수로 구성되어 있다.

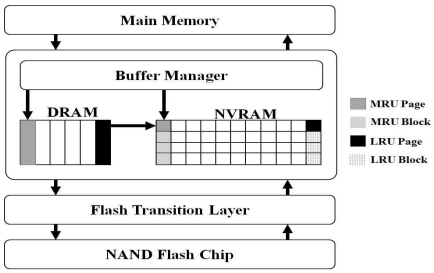

그림 1. 제안 시스템 구조도
Fig. 1 Structure of proposed system

쓰기 빈도 F_{writes}와 연속 페이지 쓰기 평균값 W_{eval}은 본 논문에서 제안하는 기법을 위해 측정되는 값이다. F_{writes}는 이전 요청의 종류와 현재 요청의 종류가 같지 않으면 최소값 1에서부터 증가하는 값으로 전체 표본 구간에서 나타나는 연속된 쓰기의 발생 빈도를 의미한다. W_{eval}은 한 페이지에서 나타나는 연속된 쓰기 페이지의 전체 크기를 연속된 쓰기가 나타낸 빈도 F_{writes}로 나눈 값으로 아래와 같은 수식 2로 나타낼 수 있다.

$$W_{eval} = \frac{\sum_{n=0}^{I_{length}} R(n)_{size}}{F_{write}} \ (\text{if } R(n)_{op} \ is \ write) \tag{2}$$

구간의 특성에 따라 본 시스템에서는 버퍼 매니저의 판단 정책을 Balanced (P_{state}가 BAL인 경우) 상태와 Write-intensive (P_{state}가 WI인 경우) 상태로 구분하였다. 제안하는 전체 시스템의 구조는 아래 그림 1과 같이 나타낼 수 있다.

호스트 시스템에서 플래시 메모리로의 접근은 디스크 버퍼를 경유하여 이루어진다. 디스크 버퍼는 호스트 요청을 제안하는 버퍼 정책에 따라 DRAM 혹은 NVRAM의 MRU 위치에 배치한다. DRAM의 공간 부족으로 인한 이주 발생은 쓰기 참조가 발생한 더티 (Dirty) 페이지에 한해 DRAM의 LRU 위치에 있는 페이지를 NVRAM의 MRU 블록으로 이주시켜서 쓰기 버퍼가 다시 한번 참조될 확률을 증가시킨다.

2.1. DRAM 버퍼 관리 기법

DRAM 버퍼 영역의 페이지 관리 기법은 LRU를 기반으로 한 페이지 단위의 관리 알고리즘을 따른다. 알고리즘 1은 제안하는 기법에서 DRAM 버퍼를 관리하는 과정을 보여준다.

```
Ensure:  The victim page by LRU is q
Input:   R_page
Output:  null
1:  if DRAM is full then
2:      if q is dirty then
3:          q migrate to NVRAM
4:      else
5:          discard page q
6:      endif
7:  insert R_page into MRU position of DRAM
8:  endif
```

알고리즘 1. DRAM 버퍼 관리
Algorithm 1. DRAM buffer management

버퍼 매니저의 배치 영역 선정으로 DRAM 버퍼 영역에서 쓰기 연산이 발생하면 버퍼 내에 할당 가능한 공간이 존재하는지 확인한다 (line 1). DRAM 버퍼가 가득 찬 상태이면 LRU 위치에 있는 페이지를 DRAM 버퍼 영역의 여유 공간을 확보하기 위한 희생 페이지로 선정한다. 희생 페이지로 선정된 페이지가 쓰기 참조가 있었던 더티 페이지라면 NVRAM 버퍼 영역으로의 이주를 수행한다 (line 2-3). 그렇지 않고 읽기 참조만 있었던 페이지의 경우 버퍼 영역에서 해당 값을 파기하여 신규 공간을 확보한다 (line 5). 충분한 공간이 확보된 상태이면 DRAM 버퍼 영역의 MRU 위치에 입력된 페이지를 삽입한다 (line 7). 제안 알고리즘은 쓰기 요청을 NVRAM 버퍼 영역으로 이주시킴으로써 버퍼에서 최대한 오래 보존한다. 버퍼 영역 접근에 의한 최신 페이지의 갱신을 통해 공간 지역성과 시간 지역성을 고려하여 버퍼 적중률을 상승시킨다.

2.2. NVRAM 버퍼 관리 기법

NVRAM 버퍼 영역에 할당된 페이지는 플래시 메모리에 대한 순차 쓰기 성능을 강화하기 위해 논리적으로 블록을 할당하며 블록 단위로 LRU 리스트를 관리한다. 논리적 블록의 크기는 플래시 메모리와 같은 크기로 할당한다.

알고리즘 2는 버퍼 매니저에 의해 NVRAM으로 직접 쓰기가 요청되거나 DRAM 버퍼 영역에서의 이주로 인해 NVRAM 버퍼 영역에 페이지 쓰기가 발생했을 때의 동작을 나타낸 것이다.

NVRAM 버퍼 영역에 페이지 쓰기가 발생하면 해당 페이지가 삽입될 블록 번호 rbn를 계산한다. 희생 블록은 LRU 영역에 있는 블록으로 선정하고, 버퍼 플러시 플래그를 false로 설정하여 요청 당시

```
Ensure:  The victim block by LRU is vb, the logical block number of R_page is rbn, the logical block object in NVRAM is block[].
Input:   R_page
Output:  null
1:  flush ← false;
2:  if block[rbn] exist in disk buffer & it has no free space then
3:      vb ← block[rbn];
4:      flush ← true
5:  endif
6:  if flush then // NVRAM is full
7:    for p in vb
8:      if p is dirty then
9:        write page p to Storage
10:     else
11:       discard page p
12:     endif
13:   end
14:   for p in DRAM
15:     set logical block number of p as pn
16:     if pn = rbn & p is dirty then
17:       write page p to Storage
18:     endif
19:   end
20: endif
21: if block[rbn] does not exist in disk buffer then
22:   //allocate free block for rbn and //set to MRU position
      AllocateFreeBLK(rbn);
23: endif
24: insert R_page into block[rbn];
```

알고리즘 2. NVRAM 버퍼 관리
Algorithm 2. NVRAM buffer management

에는 플러시 여부를 판단하지 않는다 (line 1). 삽입되는 페이지에 대한 논리 블록인 block[rbn]이 버퍼 내에 존재하면서, 해당 블록이 같은 논리적 블록을 갖는 페이지들로 가득 찼을 경우, 희생 블록을 입력 페이지가 속하는 블록 block[rbn]으로 설정하고 플러시 플래그를 true로 변경하여 플러시가 이루어지도록 한다 (line 2-5). 플러시 플래그가 설정되었을 경우 LRU 위치에 해당한 블록 혹은 block[lbn]으로 변경된 희생 블록이 플러시 된다. 플러시 과정은 블록 내 더티 페이지들을 플래시 메모리에 쓰는 연산의 수행이다 (line 11-18). 클린 페이지들에 대해서는 버퍼 영역에서 파기한다 (line 11). 또한, 희생 블록과 같은 논리 블록 번호를 가지는

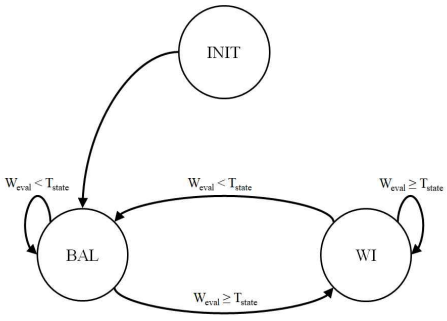

그림 2. 페이지 배치 정책 상태 다이어그램
Fig. 2 Page placement policy status diagram

DRAM의 더티 페이지들을 같이 플러시 하여 스토리지에 순차 쓰기를 수행한다 (line 14-18). 입력될 페이지가 속하는 논리적 블록이 버퍼 내에 존재하지 않으면 블록을 할당해주고 (line 22) 요청 페이지를 블록 내부에 배치한다 (line 24). NVRAM 버퍼 영역을 제안된 기법으로 관리함으로써 버퍼에서 논리적으로 같은 블록에 속한 페이지들을 같은 크기로 구성된 플래시 메모리의 블록에 그대로 쓰기 연산을 수행할 수 있다. 공간 지역성으로 이후 플래시 메모리에서 가비지 컬렉션이 호출될 때 블록 내의 인접한 페이지가 무효 페이지일 가능성을 높여 에너지 효율성을 증가시키고 플래시 읽기 연산의 순차 접근 성능을 향상할 수 있다.

2.3. 페이지 배치 전략

효율적인 하이브리드 버퍼 영역의 활용을 위해 버퍼 매니저는 2OS 테이블에 기록된 값을 사용하여 버퍼 정책을 결정한다. 버퍼 정책은 통상적인 상태에서 사용되는 Balanced 상태와 쓰기 요청이 많을 때 사용하는 Write-Intensive 상태가 있다. 다음의 그림 2는 제안 기법의 배치 전략에 대한 상태 다이어그램을 나타낸 것이다.

Balance 상태에서는 워크로드 특성에 대한 선행 분석을 통해 사전 정의된 페이지의 크기를 기준으로 순차적 페이지 접근에 대한 사전 임계치를 활용하여 특정 구간에서의 호스트 요청이 임의 접근 혹은 순차 접근 가운데 하나의 경향을 보인다고 가정한다. 요청 페이지가 순차 임계치보다 작으면 해당 요청은 임의 접근이라 판단한다. 이 경우 해당 페이지를 DRAM 버퍼 영역에 할당하며, 반대로 그 이상일 경우는 NVRAM에 할당한다. 임의 페이지 접근을 DRAM 버퍼 영역에 할당함으로써 DRAM에서의 국소적 영역에 대한 재참조를 집중시켜 성능을 향상시킨다. 한편, 큰 단위의 순차 페이지 접근을 NVRAM 버퍼 영역에 배치하여 DRAM의 공간 활용률을 증가시키고 NVRAM에서 순차 접근의 재참조가 이루어지도록 한다. 이를 통해 집적도와 속도 측면에서 차이가 나는 서로 다른 형태의 메모리 소자 특성에 맞는 유리한 연산이 가능하도록 한다.

반면, Write-Intensive 상태에서는 호스트 요청의 종류에 따라 DRAM 버퍼 영역과 NVRAM 버퍼 영역에 구분하여 페이지를 배치한다. 쓰기 요청은 DRAM 버퍼 영역에 할당하여 희생 블록으로 선정되더라도 NVRAM 버퍼 영역으로 이주하여 재참조될 가능성을 높이고, 읽기 요청은 NVRAM 버퍼 영역에 배치하여 DRAM 및 NVRAM 버퍼 영역 전체에서 읽기 버퍼를 수행하면서도 쓰기 적중률이 향상되도록 구성한다.

표 3. 메모리 특성
Table 3. Memory Characteristics

Type	Latency/Energy per page	
DRAM	read	15 ns / 0.1 nJ
	write	15 ns / 0.1 nJ
	static	1 W (per GB)
NVRAM	read	11 ns / 0.1 nJ
	write	27 ns / 0.2 6nJ
	static	0.1 W (per GB)
NAND	read	21 us / 1.5 µJ
	write	200 us / 4 µJ
	erase (block)	1.2 ms / 34 µJ

Ⅳ. 성능 평가

1. 실험 환경

제안하는 시스템 구조와 알고리즘의 성능을 평가하기 위해 워크로드 기반의 버퍼 시뮬레이터를 구현하여 실험하였다. 실험을 위한 시스템 구성은 64GB의 플래시 메모리 스토리지에 32MB의 버퍼를 구성하여 수행하였으며 한 페이지는 4KB, 한 블록은 64개의 페이지로 구성되어 있다. FTL은 블록 단위로 매핑 테이블을 관리하고 가장 기본적인 GA를 이용해 가비지 컬렉션을 수행한다. DRAM 버퍼는 페이지 단위로 관리되며 NVRAM 버퍼는 플래시 메모리와 같은 개수의 페이지를 갖는 논리적 블록으로 구성된다. 평가 지표를 위한 메모리 소자는 90nm 공정으로 구성된 DRAM, STT-MRAM을 가정하여 아래의 표 3에 나타내었다 [9-12].

표 4. 워크로드 특성
Table 4. Workload characteristics

trace	R/W ratio	T_{state}	T_{read}	T_{write}
Financial	57:43	125	512	600
online	35:64	160	271	38
web-vm	28:72	211	700	2500
webmail	26:73	230	2500	3500

비교 대상으로는 기본적인 관리 기법인 CLOCK과 LRU 기법을 활용하며, 모든 버퍼 영역을 페이지 단위로 관리한다. 반면에 하이브리드 디스크 버퍼 관리 기법인 CLOCK-DNV와 WAP-LRU는 DRAM 버퍼 영역과 NVRAM 버퍼 영역을 1:4로 배분하여 실험하였다 [6].

본 논문에서는 에너지 소모와 수명 증가를 검증하기 위해 쓰기 비중이 높은 워크로드를 선택하여 사용하였으며, 표 4는 사용된 각 워크로드의 특성과 제안 기법 적용을 위한 파라미터이다.

T_{state}는 각 워크로드에서 나타나는 연속된 쓰기 페이지의 상위 25% 구간의 값이며, T_{read}, T_{write} 값은 특정 구간별 단일 요청으로 발생한 최대 페이지 요구 개수로 중앙값과 상위 25% 값의 평균값으로 계산하여 활용하였다.

2. 실험 결과

본 논문에서 제안하는 WAP-LRU는 분석된 워크로드를 기반으로 하이브리드 버퍼 메모리에 요청을 분산하여 배치함으로써 에너지 소모 감소와 플래시 메모리의 성능 향상을 기대한다.

전체 에너지는 동적 에너지와 정적 에너지의 합산으로 계산하였다. 식 (3)은 이를 나타낸 것이다.

$$Energy\ Consumtion = E_{dynamic} + E_{static} \tag{3}$$

동적 에너지 소모량은 각각의 메모리 소자가 가지는 읽기/쓰기/삭제 연산의 횟수와 연산 에너지의 곱으로 구성되며 식 (4)를 이용해 계산하였다.

$$\begin{aligned} &E_{dynamic} = O_{(rw)} + O_{erase} \\ &O_{(rw)} = O_{DRAM(rw)} + O_{NVRAM(rw)} + O_{NAND(rw)} \\ &O_{DRAM(rw)} = (Num_{DRAM(rw)} * Energy_{DRAM(rw)}) \\ &O_{NVM(rw)} = (Num_{NVM(rw)} * Energy_{NVM(rw)}) \\ &O_{NAND(rw)} = (Num_{NAND(rw)} * Energy_{NAND(rw)}) \\ &O_{erase} = (Num_{NAND(erase)} * Energy_{NAND(erase)}) \end{aligned} \tag{4}$$

정적 에너지 소모량은 버퍼로 사용되는 두 메모리 소자의 대기 전력을 연산을 수행한 시간으로 곱한 값이다. 플래시 메모리는 같은 조건을 사용하므로 생략하였으며 아래의 식 (5)로 계산하였다.

$$\begin{aligned} E_{static} = (&Size_{DRAM} \times Energy_{static_{(DRAM)}} + \\ &Size_{NVRAM} \times Energy_{static_{(NVRAM)}}) \times Duration \end{aligned} \tag{5}$$

그림 3은 LRU, CLOCK, CLOCK-DNV, WAP-LRU를 사용하였을 때 가비지 컬렉션의 횟수를 LRU를 기준으로 정규화하여 나타낸 것이다. 모든 경우에 CLOCK 알고리즘이 가장 높은 가비지 컬렉션 호출을 수행하였고 쓰기 연산 비율이 높고 페이지 크기 분포의 편차가 크지 않은 워크로드에서 제안 기법이 평균 63.1%, 최대 99.7%의 가비지 컬렉션 감소율을 보였다. 반면, 페이지 크기의 분포가 극단으로 큰 경향이 있는 webmail 워크로드에서는 LRU나 CLOCK에 비교해서는 적은 가비지 컬렉션 횟수를 보여주지만, CLOCK-DNV보다는 32.1% 많은 가비지 컬렉션을 수행하였다 [6].

그림 4는 가비지 컬렉션과 디스크 버퍼 이주로 인해 스토리지에 쓰기가 발생하는 비율을 LRU 기법을 기준으로 정규화하여 나타낸 그래프이다. 평균적으로 39.5%의 쓰기가 감소하였고 CLOCK-DNV 기법과 비교하면 14.3% 감소한 플래시 메모리 쓰기 횟수를 확인할 수 있었다. webmail 워크로드는 가비지 컬렉션 호출 증가 때문에 플래시 스토리지로의 쓰기 접근이 증가했음을 확인할 수 있다 [6].

그림 5-7은 에너지 관점에서 제안 알고리즘의 성능 개선을 나타낸 것이다. 그림 5의 전체 에너지 소모 그래프에서 DRAM을 단일 디스크 버퍼로 사용한 기법에 비교해 하이브리드 구성은 평균 53.4%, 최대 74.1%의 에너지 소모가 줄어든다. webmail의 경우와 같이 제안 알고리즘이 불리한 워크로드 환경에서도 에너지 소모의 폭은 1%로 크지 않음을 확인할 수 있다.

그림 6은 동적 에너지의 소모 비율을 비교한 그래프이다. NVRAM의 높은 동적 에너지 소모로 인해 동적 에너지 그래프에서는 LRU 혹은 CLOCK과 같이 DRAM만 사용할 경우 적은 에너지를 사용하는 것을 확인할 수 있다. 다만, Financial의 경우는 읽기 연산의 비율이 높아 동적 에너지 소모가 낮으므로 오히려 유리한 결과가 도출되었다.

모든 워크로드에 대해 NVRAM 비율에 따른 정적 에너지와 동적 에너지의 평균적인 비율을 그림 7에 나타냈다. NVRAM을 0%에서 100%까지 25%씩 비율을 증가시키면서 실험을 진행하였으며, 전체 에너지 소모량은 평균 21.8%씩 선형적으로 감소하

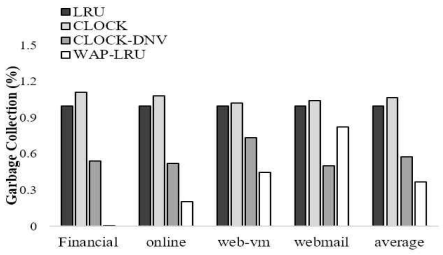

그림 3. 가비지 컬렉션 횟수
Fig. 3 The number of garbage collections

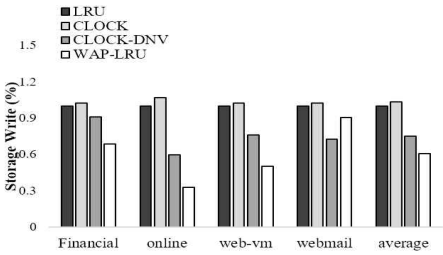

그림 4. 스토리지 쓰기 횟수
Fig. 4 The number of write to storage

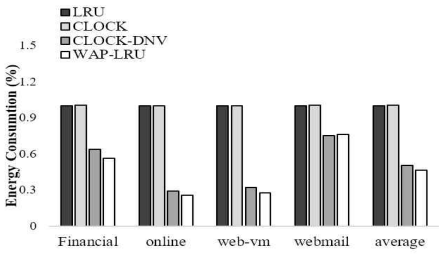

그림 5. 전체 에너지 소모율
Fig. 5 Total energy consumption

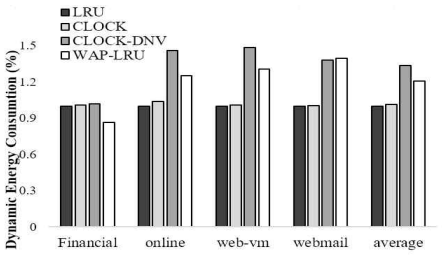

그림 6. 동적 에너지 소모율
Fig. 6 Dynamic energy consumption

나 동적 에너지의 차이는 3% 내외이다. 전체 버퍼 영역을 NVRAM으로 구성하면 (100%), 동적 에너지가 차지하는 비율이 26.1%로 높아지지만, 전체 소모 에너지는 가장 낮다. 따라서 정적 에너지 소모량이 낮은 NVRAM의 비중을 증가시키면 연산 부담이 늘어나더라도 전체 에너지에서 정적 에너지 감소로 얻는 이득이 확연하다.

V. 결 론

본 연구에서는 워크로드 분석을 기반으로 하여 메모리와 작업 특성에 맞게 디스크 버퍼에서의 페이지 배치 위치를 결정함으로써 버퍼 접근의 고효율화를 추구하였다. NVRAM 중 하나인 STT-MRAM을 DRAM으로 구성된 디스크 버퍼의 일부 영역을 대체하도록 구성하고 각 소자의 정적 에너지 손실과 동적 에너지 손실을 계산한 결과 제안된 기법에서 전체적인 에너지 소모율과 가비지 컬렉션 호출을 통한 플래시 메모리의 삭제 및 쓰기 연산 감소를 확인하였다.

향후 연구 방향으로, 제안 기법의 배치 전략 판단을 선행 분석이 아닌 동적으로 실시간 환경에 맞춰 처리할 수 있도록 알고리즘을 개선할 예정이다.

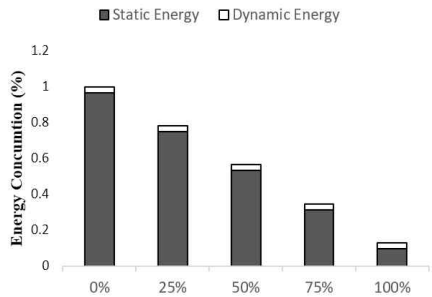

그림 7. NVRAM 비율에 따른 에너지 소모량
Fig 7. Energy consumption with respect to NVRAM ratio

References

[1] H. Kim, S. Lee, "A New Flash Memory Management for Flash Storage System," Proceedings of Computer Software and Applications Conference, pp. 284-289, 1999.

[2] H. Jo, J.U. Kang, S.Y. Park, J-S. Kim, J. Lee, "FAB: Flash Aware Buffer Management Policy for Portable Media Players," IEEE Transactions on Consumer Electronics, Vol.

52, No. 2, pp. 485-493, 2006.

[3] B. Debnath, S. Subramanya, D. Du, D. J. Lilja, "Large Block CLOCK (LB-CLOCK): A Write Caching Algorithm for Solid State Disks," Proceedings of Modeling, Analysis and Simulation of Computer and Telecommunication Systems, pp. 1-9, 2009.

[4] S.Y. Lee, H.K. Bahn, S. H. Noh, "Design of a Page Replacement Policy for Hybrid PCM and DRAM Memory," Journal of KIISE: Computer Systems and Theory, Vol. 39, No. 2, pp. 111-118 2012 (in Korean).

[5] Q. Wei, C. Chen, J. Yang, "CBM: A Cooperative Buffer Management for SSD," Proceedings of Mass Storage Systems and Technologies, pp. 1-12, 2014.

[6] D. Kang, S. Han, Y. Kim, Y. Eom, "CLOCK-DNV: A Write Buffer Algorithm for Flash Storage Devices of Consumer Electronics," IEEE Transactions Consumer Electronics, Vol. 63, no. 1, pp. 85 - 91, 2017.

[7] Z. Sun, B. Xiuyuan, H. Li, W.F. Wong, Z.L. Ong, X. Zhu, W. Wu, "Multi Retention Level STT-RAM Cache Designs With a Dynamic Refresh Scheme," Proceedings of Microarchitecture, pp. 328-338 2011.

[8] X. Dong, C. Xu, N. Jouppi, Y. Xie, "NVSim: A Circuit-Level Performance, Energy, and Area Model for Emerging Nonvolatile Memory," Proceedings of Emerging Memory Technologies, pp. 15-50, 2014.

[9] M.K. Qureshi, V. Srinivasan, J.A. Rivers, "Scalable High Performance Main Memory System Using Phase-Change Memory Technology," Proceedings of ACM SIGARCH Computer Architecture, pp. 24-33, 2009.

[10] S. Pelley, T.F. Wenisch, B.T. Gold, B. Bridge, "Storage Management in the NVRAM Era," Proceedings of the VLDB Endowment, Vol. 7, No. 2, pp. 121-132, 2013.

[11] Yuan Xie, "Emerging Memory Technologies Design, Architecture and Applications," Chapter 1, Springer, 2014

[12] H.-S. P. Wong, C. Ahn, J. Cao, H.-Y. Chen, S. W. Fong, Z. Jiang, C. Neumann, S. Qin, J. Sohn, Y. Wu, S. Yu, X. Zheng, H. Li, J. A. Incorvia, S. B. Eryilmaz, K. Okabe, "Stanford Memory Trends," Avaliable: https://nano.stanford.edu/stanford-memory-trends, 2016.

[13] N. Agrawal, V. Prabhakaran, T. Wobber, J.D. Davis, M.S. Manasse, R. Panigrahy, "Design tradeoffs for SSD performance," Proceedings of USENIX Annual Technical Conference on Annual Technical Conference, Vol. 57, pp. 57-70, 2008.

Kyung Min Kim (김 경 민)

He received a B.S. degree in Department of Computer Engineering from Yeungnam University, Korea in 2017. He is currently a M.S. student in Department of Computer Engineering at Yeungnam University. His current research interests include s/w architecture and non-volatile memory.

Email: bl43a@ynu.ac.kr

Jun-Hyeong Choi (최 준 형)

He received a B.S. degree in Department of Computer Engineering from Yeungnam University College, Daegu, Korea in 2013, a M.S. degree in Department of Computer Engineering from Yeungnam University, Korea, in 2016, respectively. He is currently a Ph.D. student in Department of Computer Engineering at Yeungnam University. His current research interests include distributed systems and non-volatile memory.

Email: runchoice@ynu.ac.kr

Jong Wook Kwak (곽 종 욱)

He received a B.S. degree in Computer Engineering from Kyungpook National University, Taegu, 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. He is currently an associate professor in the Department of Computer Engineering, Yeungnam University. His research interests include advanced processor architecture, low-power mobile embedded system, and high performance parallel computing.

Email: kwak@yu.ac.kr