



(19) 대한민국특허청(KR)
(12) 공개특허공보(A)

(11) 공개번호 10-2017-0109110
(43) 공개일자 2017년09월28일

(51) 국제특허분류(Int. Cl.)
G06F 12/08 (2016.01) G06F 13/16 (2006.01)
G06F 3/06 (2006.01)
(52) CPC특허분류
G06F 12/0871 (2013.01)
G06F 13/1673 (2013.01)
(21) 출원번호 10-2016-0032040
(22) 출원일자 2016년03월17일
심사청구일자 2016년03월17일

(71) 출원인
이화여자대학교 산학협력단
서울특별시 서대문구 이화여대길 52 (대현동, 이화여자대학교)
(72) 발명자
반효경
서울특별시 관악구 관악로30길 12 107동 2001호
(봉천동, 우성아파트)
김도희
서울특별시 은평구 갈현로3다길 15 101동 503호
(신사동, 라이프씨티아파트)
(74) 대리인
김인철

전체 청구항 수 : 총 10 항

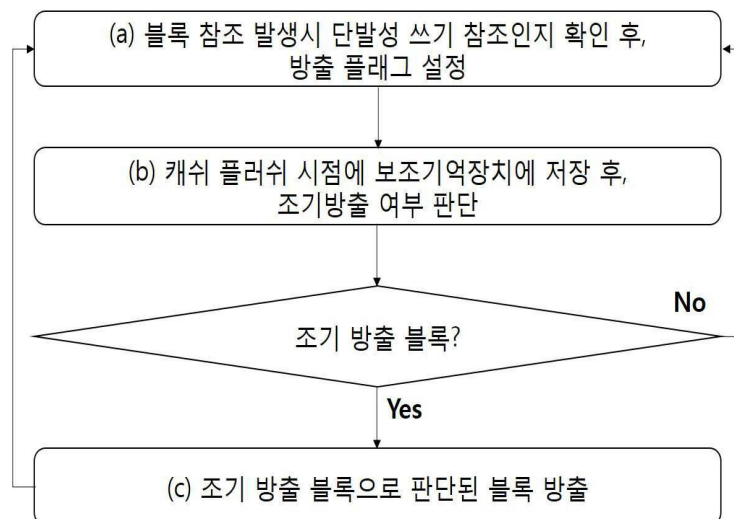
(54) 발명의 명칭 단발성 쓰기 특성을 고려한 모바일 기기의 버퍼캐쉬 관리방법

(57) 요약

본 발명은 단발성 쓰기 특성을 고려한 모바일 기기의 버퍼캐쉬 관리방법에 관한 것으로, (a) 버퍼캐쉬 관리부가 플러시 반영 시점 이전까지 해당 블록이 1회 쓰기 참조에 의한 조기 방출 블록 여부를 판단하는 단계; (b) 버퍼캐쉬 관리부가 해당 블록이 조기 방출 블록이라고 판단한 경우, 캐쉬 플러시(flush) 시점에 보조 기억장치에 블록의 데이터를 저장하는 단계; 및 (c) 버퍼캐쉬 관리부가 캐쉬 플러시(flush) 시점에 조기 방출 블록으로 판단된 블록을 방출하는 단계를 포함한다.

이와 같은 본 발명은, 공간 효율성을 높이는 버퍼캐쉬 관리방법을 제공하고, 스마트폰 등의 모바일 기기의 파일 시스템의 성능을 향상시키며, 모바일 기기의 앱 실행에 필요한 메모리 용량을 대폭 감소시켜 배터리 소모를 줄일 수 있는 버퍼캐쉬 관리방법을 제공한다.

대표도 - 도1



(52) CPC특허분류

G06F 3/064 (2013.01)

G06F 2212/1044 (2013.01)

G06F 2212/6022 (2013.01)

이 발명을 지원한 국가연구개발사업

과제고유번호 2011-0028825

부처명 미래창조과학부

연구관리전문기관 한국연구재단

연구사업명 기초연구사업-중견연구자지원사업-도약연구(전략)

연구과제명 프로그램 특성 분석에 기반한 스마트 디바이스용 임베디드 운영체제 설계

기 여 율 50/100

주관기관 이화여자대학교 산학협력단

연구기간 2011.09.01 ~ 2016.08.31이 발명을 지원한 국가연구개발사업

과제고유번호 1711025723

부처명 미래창조과학부

연구관리전문기관 한국산업기술평가관리원

연구사업명 SW컴퓨팅산업원천기술개발사업

연구과제명 차세대 메모리기반의 스마트디바이스용 임베디드시스템 소프트웨어원천기술개발

기 여 율 50/100

주관기관 한양대학교 산학협력단

연구기간 2012.06.01 ~ 2017.05.31

명세서

청구범위

청구항 1

- (a) 버퍼캐쉬 관리부가 캐쉬 플러쉬 시점 이전까지 블록 참조 발생시 해당 블록이 단발성 쓰기 참조인지 여부를 확인하고 방출 플래그를 설정하는 단계;
- (b) 버퍼캐쉬 관리부가 해당 블록을 캐쉬 플러쉬(flush) 시점에 보조 기억장치에 블록의 데이터를 저장한 후, 조기방출 여부를 판단하는 단계; 및
- (c) 버퍼캐쉬 관리부가 조기 방출 블록으로 판단된 블록을 방출하는 단계를 포함하는 것을 특징으로 하는 단발성 쓰기 특성을 고려한 모바일 기기의 버퍼캐쉬 관리방법.

청구항 2

청구항 1에 있어서,

- (a) 단계는,

블록 요청이 있는 경우, 버퍼캐쉬 관리부가 해당 블록이 버퍼 캐쉬 에 있는지 여부를 판단하는 단계; 및

버퍼캐쉬 관리부가 버퍼 캐쉬에 해당 블록이 있고 캐쉬 플러쉬(flush) 시점에 전까지 해당 블록 1회 쓰기 참조인지 여부를 판단하는 단계를 포함하는 것을 특징으로 하는 단발성 쓰기 특성을 고려한 모바일 기기의 버퍼캐쉬 관리방법.

청구항 3

청구항 1에 있어서,

블록 요청이 있고 해당 블록이 버퍼 캐쉬에 없고 공간이 없는 경우, 최근 사용 횟수 방식(LRU)에 의해 블록을 교체하는 단계를 포함하는 것을 특징으로 하는 단발성 쓰기 특성을 고려한 모바일 기기의 버퍼캐쉬 관리방법.

청구항 4

청구항 1에 있어서,

- 상기 (a) 단계는,

버퍼캐쉬 관리부가 캐쉬 플러쉬(flush) 시점 이전까지 해당 블록이 1회성 쓰기 참조가 있는 경우, 해당 블록에 방출 플래그를 설정하는 단계를 포함하는 것을 특징으로 하는 단발성 쓰기 특성을 고려한 모바일 기기의 버퍼캐쉬 관리방법.

청구항 5

청구항 1에 있어서,

상기 캐쉬 플러쉬(flush) 시점은, 3초 내지 8초 중 어느 한 시간 구간 동안 주기적으로 반복되는 것을 특징으로 하는 단발성 쓰기 특성을 고려한 모바일 기기의 버퍼캐쉬 관리방법.

청구항 6

청구항 1에 있어서,

- 상기 (b) 단계에서 조기 방출 블록 여부를 판단하는 단계는,

해당 블록이 캐쉬 플러쉬(flush) 시점 전까지 쓰기 참조가 1회의 단발성 여부를 판단하는 단계;

캐쉬 플러쉬 (flush) 시점까지 해당 블록에 대해 읽기 참조가 있는지 여부를 판단하는 단계; 및

해당 블록이 캐쉬 플러쉬(flush) 시점 전까지 쓰기 참조가 1회의 단발성이고, 읽기 참조가 없는 경우에 조기 방

출 블록으로 판단하는 단계를 포함하는 것을 특징으로 하는 단발성 쓰기 특성을 고려한 모바일 기기의 버퍼캐쉬 관리방법.

청구항 7

청구항 1에 있어서,

상기 (b) 단계에서 조기 방출 블록 여부를 판단하는 단계는,

해당 블록이 캐쉬 플러쉬(flush) 시점 전까지 쓰기 참조가 1회의 단발성 여부를 판단하는 단계;

캐쉬 플러쉬(flush) 시점까지 해당 블록에 대해 읽기 참조가 있는지 여부를 판단하는 단계;

캐쉬 플러쉬(flush) 시점까지 해당 블록이 히스토리 버퍼에 존재하는지 여부를 판단하는 단계; 및

캐쉬 플러쉬(flush) 시점 전까지 해당 블록이 쓰기 참조가 1회의 단발성이고, 읽기 참조가 없고, 히스토리 버퍼에 존재하지 않는 경우에 조기 방출 블록으로 판단하는 단계를 포함하는 것을 특징으로 하는 단발성 쓰기 특성을 고려한 모바일 기기의 버퍼캐쉬 관리방법.

청구항 8

청구항 1에 있어서,

상기 (b) 단계는,

캐쉬 플러쉬(flush) 시점에 해당 블록의 데이터를 보조 기억 장치에 저장한 후, 상기 데이터 블록의 설정을 해제하는 단계를 포함하는 것을 특징으로 하는 단발성 쓰기 특성을 고려한 모바일 기기의 버퍼캐쉬 관리방법.

청구항 9

청구항 1에 있어서,

상기 (c) 단계는,

방출 블록으로 판단된 블록을 히스토리 버퍼에 저장하는 단계를 포함하는 것을 특징으로 하는 단발성 쓰기 특성을 고려한 모바일 기기의 버퍼캐쉬 관리방법.

청구항 10

하드웨어와 결합되어,

(a) 버퍼캐쉬 관리부가 캐쉬 플러쉬 시점 이전까지 블록 참조 발생시 해당 블록이 단발성 쓰기 참조인지 여부를 확인하고 방출 플래그를 설정하는 단계;

(b) 버퍼캐쉬 관리부가 해당 블록을 캐쉬 플러쉬(flush) 시점에 보조 기억장치에 블록의 데이터를 저장한 후, 조기방출 여부를 판단하는 단계; 및

(c) 버퍼캐쉬 관리부가 조기 방출 블록으로 판단된 블록을 방출하는 단계를 실행시키기 위하여 컴퓨터로 판독 가능한 매체에 저장된 컴퓨터프로그램.

발명의 설명

기술 분야

[0001] 본 발명은 버퍼캐쉬 관리방법에 관한 것으로, 보다 상세하게는 공간 효율성을 높이고, 파일시스템의 성능을 향상시키며, 배터리 소모를 줄일 수 있는 단발성 쓰기 특성을 고려한 모바일 기기의 버퍼캐쉬 관리방법에 관한 것이다.

배경 기술

[0002] 버퍼캐쉬(buffer cache)는 대용량 서버에서 모바일 디바이스에 이르는 다양한 컴퓨터 시스템에서 느린 스토리지의 접근 속도를 완충하기 위해 널리 사용된다. 버퍼캐쉬는 최근에 사용된 파일 데이터를 메모리에 보관하여 동

일 데이터에 대한 재요청시 스토리지 접근 없이 해당 데이터를 직접 서비스하여 성능 향상을 도모한다.

[0003] 한편, 스마트폰에서는 갑작스런 물리적 충격 혹은 전원 오류 시 파일시스템의 일관성이 깨어지는 문제가 발생할 수 있다. 이를 방지하기 위해 저널링(혹은 로깅)을 사용하여 파일 데이터의 수정본을 관리한다.

[0004] 저널링은 파일 데이터를 수정하기 전에 복사본을 만들어 둬으로써 수정 도중에 크래쉬가 발생하더라도 직전 버전으로의 복원을 가능하게 한다. 이 과정에서 재사용되지 않을 데이터가 버퍼캐쉬에 추가로 생성되는 비효율성이 발생한다. 예를 들어 스마트폰의 버퍼캐쉬에서 파일 데이터 A_{data} 가 수정될 경우 복사본 파일 $A_{journal}$ 을 먼저 생성한 후 A_{data} 를 수정한다. 이렇게 되면 $A_{journal}$ 은 낡은 데이터이고 최신 데이터는 A_{data} 이므로 향후 모든 읽기 쓰기 요청은 A_{data} 에 대해 발생한다.

[0005] 한편, 버퍼캐쉬에 존재하던 $A_{journal}$ 과 A_{data} 는 스토리지 반영(flush) 연산에 의해 스토리지에 기록되어 갑작스런 전원 오류에 대비한다. 스토리지 기록 후 최신본 A_{data} 는 계속 사용될 수 있으나 낡은 데이터인 $A_{journal}$ 은 사용 가능성이 없으므로 버퍼캐쉬에 남아 있는 동안 캐쉬 공간이 낭비되는 문제점이 있다. 그러나, 버퍼캐쉬 관점에서는 $A_{journal}$ 이 A_{data} 의 낡은 버전을 인지할 수 없으므로 종래의 버퍼캐쉬 시스템은 두 데이터 모두를 버퍼캐쉬에 유지하는 문제점이 있다.

선행기술문헌

특허문헌

[0006] (특허문헌 0001) 대한민국 공개특허공보 제10-2014-0137839호(공개일자: 2014년12월03일)

(특허문헌 0002) 대한민국 등록특허공보 제10-1288276호(등록일자: 2013년07월15일)

발명의 내용

해결하려는 과제

[0007] 본 발명에 따른 단발성 쓰기 특성을 고려한 모바일 기기의 버퍼캐쉬 관리방법은 다음과 같은 해결과제를 가진다.

[0008] 첫째, 본 발명은 모바일 기기의 특성으로 나타나는 단발성 쓰기 참조의 파일 데이터의 경우 스토리지 반영 시점에 버퍼캐쉬에서 방출시켜 공간 효율성을 높이는 버퍼캐쉬 관리방법을 제공하고자 함이다.

[0009] 둘째, 본 발명은 스마트폰 등의 모바일 기기의 파일시스템의 성능을 향상시킬 수 있는 버퍼캐쉬 관리방법을 제공하고자 함이다.

[0010] 셋째, 본 발명은 모바일 기기의 앱 실행에 필요한 메모리 용량을 대폭 감소시켜 배터리 소모를 줄일 수 있는 버퍼캐쉬 관리방법을 제공하고자 함이다.

[0011] 본 발명의 해결과제는 이상에서 언급한 것들에 한정되지 않으며, 언급되지 아니한 다른 해결과제들은 아래의 기재로부터 당업자에게 명확하게 이해될 수 있을 것이다.

과제의 해결 수단

[0012] 상술한 과제를 해결하고자 하는 본 발명의 제1 특징은, (a) 버퍼캐쉬 관리부가 캐쉬 플러쉬 시점 이전까지 블록 참조 발생시 해당 블록이 단발성 쓰기 참조인지 여부를 확인하고 방출 플래그를 설정하는 단계; (b) 버퍼캐쉬 관리부가 해당 블록을 캐쉬 플러쉬(flush) 시점에 보조 기억장치에 블록의 데이터를 저장한 후, 조기방출 여부를 판단하는 단계; 및 (c) 버퍼캐쉬 관리부가 조기 방출 블록으로 판단된 블록을 방출하는 단계를 포함한다.

[0013] 여기서, (a) 단계는, 블록 요청이 있는 경우, 버퍼캐쉬 관리부가 해당 블록이 버퍼 캐쉬에 있는지 여부를 판단하는 단계; 및 버퍼캐쉬 관리부가 버퍼 캐쉬에 해당 블록이 있고 캐쉬 플러쉬(flush) 시점에 전까지 해당 블록 1회 쓰기 참조인지 여부를 판단하는 단계를 포함하는 것이 바람직하다.

[0014] 또한, 블록 요청이 있고 해당 블록이 버퍼 캐쉬에 없고 공간이 없는 경우, 최근 사용 횟수 방식(LRU)에 의해 블

록을 교체하는 단계를 포함하는 것이 바람직하고, 상기 (a) 단계는, 버퍼캐쉬 관리부가 캐쉬 플러쉬(flush) 시점 이전까지 해당 블록이 1회성 쓰기 참조가 있는 경우, 해당 블록에 방출 플래그를 설정하는 단계를 포함하는 것이 바람직하며, 상기 캐쉬 플러쉬(flush) 시점은, 3초 내지 8초 중 어느 한 시간 구간 동안 주기적으로 반복되는 것이 바람직하다.

[0015] 더하여, 상기 (b) 단계에서 조기 방출 블록 여부를 판단하는 단계는, 해당 블록이 캐쉬 플러쉬(flush) 시점 전까지 쓰기 참조가 1회의 단발성 여부를 판단하는 단계; 캐쉬 플러쉬(flush) 시점까지 해당 블록에 대해 읽기 참조가 있는지 여부를 판단하는 단계; 및 해당 블록이 캐쉬 플러쉬(flush) 시점 전까지 쓰기 참조가 1회의 단발성이고, 읽기 참조가 없는 경우에 조기 방출 블록으로 판단하는 단계를 포함하는 것이 바람직하다.

[0016] 또한, 상기 (b) 단계에서 조기 방출 블록 여부를 판단하는 단계는, 해당 블록이 캐쉬 플러쉬(flush) 시점 전까지 쓰기 참조가 1회의 단발성 여부를 판단하는 단계; 캐쉬 플러쉬(flush) 시점까지 해당 블록에 대해 읽기 참조가 있는지 여부를 판단하는 단계; 캐쉬 플러쉬(flush) 시점까지 해당 블록이 히스토리 버퍼에 존재하는지 여부를 판단하는 단계; 및 캐쉬 플러쉬(flush) 시점 전까지 해당 블록이 쓰기 참조가 1회의 단발성이고, 읽기 참조가 없고, 히스토리 버퍼에 존재하지 않는 경우에 조기 방출 블록으로 판단하는 단계를 포함하는 것이 바람직하다.

[0017] 또한, 상기 (b) 단계는, 캐쉬 플러쉬(flush) 시점에 해당 블록의 데이터를 보조 기억 장치에 저장한 후, 상기 데이터 블록의 설정을 해제하는 단계를 포함하는 것이 바람직하고, 상기 (c) 단계는, 방출 블록으로 판단된 블록을 히스토리 버퍼에 저장하는 단계를 포함하는 것이 바람직하다.

[0019] 그리고, 본 발명의 제2 특징은 하드웨어와 결합되어, (a) 버퍼캐쉬 관리부가 캐쉬 플러쉬 시점 이전까지 블록 참조 발생시 해당 블록이 단발성 쓰기 참조인지 여부를 확인하고 방출 플래그를 설정하는 단계; (b) 버퍼캐쉬 관리부가 해당 블록을 캐쉬 플러쉬(flush) 시점에 보조 기억장치에 블록의 데이터를 저장한 후, 조기방출 여부를 판단하는 단계; 및 (c) 버퍼캐쉬 관리부가 조기 방출 블록으로 판단된 블록을 방출하는 단계를 실행시키기 위하여 컴퓨터로 판독 가능한 매체에 저장된 컴퓨터프로그램이다.

발명의 효과

[0020] 본 발명에 따른 단발성 쓰기 특성을 고려한 모바일 기기의 버퍼캐쉬 관리방법은 다음과 같은 효과를 가진다.

[0021] 첫째, 본 발명은 모바일 기기에서 1회의 단발성 쓰기 참조 특성을 고려하여 메인 메모리의 버퍼캐쉬에서 1회의 단발성 쓰기 참조를 갖는 블록 데이터를 조기 방출시켜, 버퍼캐쉬의 공간 낭비를 유발하는 비효율성을 개선할 수 있는 버퍼캐쉬 관리방법을 제공한다.

[0022] 둘째, 본 발명은 스마트폰 등의 모바일 기기의 파일시스템의 성능을 향상시킬 수 있는 버퍼캐쉬 관리방법을 제공한다.

[0023] 셋째, 본 발명은 공간이 필요한 시점에 캐쉬 블록을 방출하는 기존의 캐쉬 블록 방출 정책과 호환성을 가지면서, 스마트폰의 메모리 용량이 커질수록 배터리 소모가 크게 증가하기 때문에, 동일한 성능을 얻기 위해 필요한 메모리 용량을 대폭 감소시켜 배터리 소모를 줄일 수 있는 버퍼캐쉬 관리방법을 제공한다.

[0024] 본 발명의 효과는 이상에서 언급한 것들에 한정되지 않으며, 언급되지 아니한 다른 효과들은 아래의 기재로부터 당업자에게 명확하게 이해될 수 있을 것이다.

도면의 간단한 설명

[0025] 도 1은 본 발명의 실시예에 따른 단발성 쓰기 특성을 고려한 모바일 기기의 버퍼캐쉬 관리방법의 흐름을 나타낸 도면이다.

도 2는 스마트폰 등의 모바일 기기에서 전체 파일 접근 중 읽기와 쓰기의 비율을 보여주는 그래프이다.

도 3은 스마트폰 등의 모바일 기기에서 쓰기 참조가 발생한 데이터 중 1번의 쓰기만 발생한 비율을 보여주는 그래프이다.

도 4는 종래의 버퍼캐쉬에서 입출력 발생의 예를 나타내는 모식도이다.

도 5는 본 발명의 실시예에 따른 버퍼캐쉬 관리방법을 이용하여 입출력 발생의 예를 나타낸 모식도이다.

도 6 내지 도 8은 캐쉬 크기의 변화에 따라 종래 캐쉬 관리기법과 제안한 본 발명의 실시예에 따른 관리방법의 캐쉬 부재율(miss ratio) 나타낸 그래프이다.

도 9는 Mix1 워크로드에서 동일한 성능을 나타내는 두 지점인 A와 B에서 기존 기법과 본 발명의 실시예에 따른 버퍼캐쉬 관리방법에 필요한 캐쉬 크기를 나타내는 그래프이다.

도 10 및 도 11은 동일한 캐쉬 성능을 나타내는 지점에 대해 기존 기법과 제안한 본 발명의 실시예에 따른 버퍼캐쉬 관리방법의 총 수행시간과 전력소모를 비교한 그래프이다.

발명을 실시하기 위한 구체적인 내용

- [0026] 이하, 첨부한 도면을 참조하여, 본 발명이 속하는 기술분야에서 통상의 지식을 가진 자가 용이하게 실시할 수 있도록 본 발명의 실시예를 설명한다. 본 발명이 속하는 기술분야에서 통상의 지식을 가진 자가 용이하게 이해할 수 있는 바와 같이, 후술하는 실시예는 본 발명의 개념과 범위를 벗어나지 않는 한도 내에서 다양한 형태로 변형될 수 있다. 가능한 한 동일하거나 유사한 부분은 도면에서 동일한 도면부호를 사용하여 나타낸다.
- [0027] 본 명세서에서 사용되는 전문용어는 단지 특정 실시예를 언급하기 위한 것이며, 본 발명을 한정하는 것을 의도하지는 않는다. 여기서 사용되는 단수 형태들은 문구들이 이와 명백히 반대의 의미를 나타내지 않는 한 복수 형태들도 포함한다.
- [0028] 본 명세서에서 사용되는 "포함하는"의 의미는 특정 특성, 영역, 정수, 단계, 동작, 요소 및/또는 성분을 구체화하며, 다른 특정 특성, 영역, 정수, 단계, 동작, 요소, 성분 및/또는 군의 존재나 부가를 제외시키는 것은 아니다.
- [0029] 본 명세서에서 사용되는 기술용어 및 과학용어를 포함하는 모든 용어들은 본 발명이 속하는 기술분야에서 통상의 지식을 가진 자가 일반적으로 이해하는 의미와 동일한 의미를 가진다. 사전에 정의된 용어들은 관련기술문헌과 현재 개시된 내용에 부합하는 의미를 가지는 것으로 추가 해석되고, 정의되지 않는 한 이상적이거나 매우 공식적인 의미로 해석되지 않는다.
- [0031] 이하에서 본 발명의 바람직한 실시예를 도면을 참조하여 상세히 설명하기로 한다.
- [0032] 도 1은 본 발명의 실시예에 따른 단발성 쓰기 특성을 고려한 모바일 기기의 버퍼캐쉬 관리방법의 흐름을 나타낸 도면이다. 도 1에 나타난 바와 같이, 본 발명의 실시예에 따른 버퍼캐쉬 관리방법은, (a) 버퍼캐쉬 관리부가 캐쉬 플러쉬 시점 이전까지 블록 참조 발생시 해당 블록이 단발성 쓰기 참조인지 여부를 확인하고 방출 플래그를 설정하는 단계; (b) 버퍼캐쉬 관리부가 해당 블록을 캐쉬 플러쉬 (flush) 시점에 보조 기억장치에 블록의 데이터를 저장한 후, 조기방출 여부를 판단하는 단계; 및 (c) 버퍼캐쉬 관리부가 조기 방출 블록으로 판단된 블록을 방출하는 단계를 포함하여 구성된다.
- [0033] 이와 같이, 스마트폰 앱의 파일참조 분석을 통해 상당 부분의 스마트폰 등의 모바일 기기의 파일 입출력 요청이 읽기보다 쓰기 위주라는 것과 이들 쓰기 중 상당 부분은 1회성 쓰기 참조라는 특성이 있고, 이러한 쓰기 참조는 SQLite와 같은 스마트폰용 데이터베이스가 저널링을 하는 과정에서 발생하는 것으로 이들이 한정된 버퍼캐쉬 공간을 낭비하는 주요 요인이 된다는 점에 착안하여, 본 발명의 실시예에서는 1번의 단발성 쓰기 참조가 이루어진 파일 데이터의 경우 스토리지 반영 시점에 버퍼캐쉬에서 방출시켜 공간 효율성을 높이는 버퍼캐쉬 관리방법을 제안한다. 여기서 모바일 기기는 스마트폰, 태블릿 PC 등 각종의 다양한 어플리케이션(앱) 프로그램을 실행시킬 수 있는 휴대용 컴퓨팅 장치이면 모두 가능하다.
- [0034] 본 발명의 실시예에서는 상술한 스마트폰 등의 모바일 기기의 파일 입출력 요청이 읽기보다 쓰기 위주이고 단발성의 1회성 쓰기 참조 특성이 있다는 점을 고려하기 위해, 대표적인 스마트폰 앱들의 파일 참조 트레이스를 추출하여 읽기 쓰기 참조의 비율 및 특성을 분석하였다.
- [0035] 도 2는 스마트폰 등의 모바일 기기에서 전체 파일 접근 중 읽기와 쓰기의 비율을 보여주는 그래프이다. 도 2에 나타난 바와 같이, 파일 데이터 접근의 상당 부분이 쓰기 참조인 것을 확인할 수 있다. 쓰기의 비율이 가장 낮은 유튜브의 경우에도 54%를 차지했으며 일부 앱은 쓰기 참조가 97%에 이르렀다.
- [0036] 도 3은 스마트폰 등의 모바일 기기에서 쓰기 참조가 발생한 데이터 중 1번의 쓰기만 발생한 비율을 보여주는 그래프이다. 도 3에 나타난 바와 같이, 1회성 쓰기 참조가 발생한 데이터의 비율은 최대 95%에 이를 정도로 큰 비

중을 차지하는 것을 확인할 수 있다. 이러한 데이터는 캐쉬에 두더라도 성능 향상에 도움이 되지 않으므로 이러한 불필요한 데이터에 의해 캐쉬 공간이 낭비되고 있음을 알 수 있다.

[0037] 이와 같은 스마트폰 등의 모바일 기기에서 나타나는 단발성 쓰기 참조 특성을 고려하여, 본 발명의 실시예는 메인 메모리의 버퍼캐쉬에서 블록 데이터의 조기 방출 방법을 제안함으로써, 스마트폰 등의 모바일 기기에서 사용되는 저널링 방식이 버퍼 캐쉬의 공간 낭비를 유발하는 비효율성을 개선하고, 스마트폰 등의 모바일 기기의 파일시스템의 성능을 향상시킬 수 있는 버퍼캐쉬 관리방법을 제공한다.

[0038] 그리고 본 발명의 실시예는 공간이 필요한 시점에 캐쉬 블록을 방출하는 기존의 캐쉬 블록 방출 정책과 호환성을 가지면서, 불필요한 캐쉬 공간의 낭비를 크게 줄여 캐쉬 공간의 효율성을 극대화시킬 수 있을 뿐만 아니라, 스마트폰의 메모리 용량이 커질수록 배터리 소모가 크게 증가하기 때문에, 동일한 성능을 얻기 위해 필요한 메모리 용량을 대폭 감소시키므로 배터리 소모를 줄이는 효과를 얻을 수 있는 장점이 있다.

[0040] 이하에서 보다 구체적으로 본 발명의 실시예에 따른 단발성 쓰기 특성을 고려한 모바일 기기의 버퍼캐쉬 관리방법의 프로세스를 상세히 설명하기로 한다.

[0042] 1회성 쓰기 참조를 고려한 버퍼캐쉬 관리방법

[0043] 캐쉬에서는 파일 데이터를 블록 단위로 관리하며, 캐쉬에 진입한 후 수정되지 않은 블록은 클린 블록(clean block), 수정된 블록은 더티 블록(dirty block)으로 구분한다. 더티 블록은 프로그램이 명시적으로 동기화(sync) 요청을 할 때, 혹은 동기화 요청이 없더라도 시스템에서 5초 주기로 스토리지에 반영(flush)하며, 반영 후 클린 상태로 바뀌게 된다.

[0044] 도 4는 종래의 버퍼캐쉬에서 입출력 발생의 예를 나타내는 모식도이다. 도 4에 나타난 바와 같이, 블록 A는 0초 지점에 쓰기 참조가 발생하여 더티 블록이 되었고, 3초 지점에는 읽기 참조가 발생하였다. 5초 시점에 주기적인 스토리지 반영 연산이 수행되어 수정본이 스토리지에 기록되며 이후 블록 A는 클린 블록이 된다. 블록 B의 경우 7초와 9초 지점에 각각 쓰기 참조가 발생했는데, 10초 지점에 스토리지 반영시 마지막으로 쓰기가 발생한 9초 지점의 내용이 스토리지에 기록된다.

[0045] 앞서 도 3에서 스마트폰 등의 모바일 기기 버퍼캐쉬에서의 쓰기 참조 분석을 통해 1회성 쓰기 참조의 비율이 높음을 확인하였다. 따라서 제안하는 본 발명의 실시예에 따른 버퍼캐쉬 관리방법은 스토리지 반영 연산이 발생할 때, 1회성 쓰기 참조만 발생한 블록을 버퍼캐쉬에서 조기 방출함으로써 캐쉬 공간의 효율을 높일 수 있게 된다.

[0046] 그리고, 본 발명의 실시예에서는 쓰기 참조가 1번만 발생했더라도 읽기 참조가 발생한 블록은 캐쉬에서 쫓아내지 않는다. 이는 읽기 참조가 발생한 블록의 경우 짧은 시간 안에 재사용 가능성이 높은 특징을 반영하여 캐쉬의 효과를 높이기 위함이다.

[0047] 도 5는 본 발명의 실시예에 따른 버퍼캐쉬 관리방법을 이용하여 입출력 발생의 예를 나타낸 모식도이다. 즉, 도 4의 종래의 버퍼캐쉬와 같은 동일한 읽기 쓰기 요청에 대해 본 발명의 실시예에 따라 버퍼캐쉬가 발생시키는 입출력의 예를 보여주고 있다.

[0048] 도 5에 나타난 바와 같이, 5초 지점에서 스토리지 반영(flush) 연산 발생 시 블록 A는 쓰기 횟수가 1번이지만 3초 지점에서 읽기 요청이 발생하였으므로 보조 기억장치와 같은 스토리지에 기록할 뿐 캐쉬에서 조기 방출하지 않고, 블록 B의 경우에는 5초 지점까지 쓰기가 1번만 발생하고 읽기 요청이 없었으므로, 5초 지점에 블록 B의 수정 내용을 스토리지로 기록한 후 캐쉬에서 방출한다.

[0049] 즉, 도 1에 나타난 바와 같이, 본 발명의 실시예에 따른 버퍼캐쉬 관리방법은, (a) 단계에서, 버퍼캐쉬 관리부가 캐쉬 플러쉬 시점 이전까지 블록 참조 발생시 해당 블록이 단발성 쓰기 참조인지 여부를 확인하고 방출 플래그를 설정하고, (b) 단계에서, 버퍼캐쉬 관리부가 해당 블록을 캐쉬 플러쉬(flush) 시점에 보조 기억장치에 블록의 데이터를 저장한 후, 조기방출 여부를 판단하고, (c) 버퍼캐쉬 관리부가 캐쉬 플러쉬(flush) 시점에 조기 방출 블록으로 판단된 블록을 방출하게 된다.

[0050] 여기서, 캐쉬 플러쉬(flush) 시점은, 3초 내지 8초 중 어느 한 시간 구간 동안 주기적으로 반복되는 것이 바람직하다. 캐쉬 플러쉬는 캐쉬에서 수정된 내용을 보조 기억장치에 반영하는 동작을 말하는 것으로, 갑작스런 크래쉬 상황에서 캐쉬 내용이 사라지는 것을 대비하기 위해 주기적으로 수행한다. 본 발명의 실시예에서는 5초를

주기로 플러쉬 시점을 설정한 예를 나타내어 설명한다.

- [0051] (a) 단계는, 해당 블록이 캐쉬 플러쉬(flush) 시점 전까지 쓰기 참조가 1회의 단발성 여부를 판단하고, 해당 블록에 대해 읽기 참조가 있는지 여부를 판단한 후 방출 플래그를 설정하게 된다.
- [0052] 즉, 도 5에 나타낸 바와 같이, 블록이 조기 방출 대상인지 아닌지를 구별하기 위해 본 발명의 실시예에서는 블록마다 방출 플래그(eviction flag)를 두어 블록의 쓰기 횟수가 1번이고 읽기 참조가 발생하지 않았는지 여부를 표시한다. 도 5에서 방출 플래그가 'o'인 경우 해당 블록은 스토리지 반영 시점에 캐쉬에서 삭제할 대상이고 'x'인 경우 그렇지 않음을 나타낸 하나의 예이다.
- [0053] 그리고, 상기 (a) 단계는 버퍼캐쉬 관리부가 캐쉬 플러쉬(flush) 시점 이전까지 해당 블록에 대한 쓰기 참조가 있는 경우, 해당 블록이 단발성 쓰기 참조인지 여부를 판단하고 방출 플래그로 태깅하는 것이 바람직하다.
- [0054] 그리고 나서, (b) 단계에서, 1회성 쓰기 참조 이후, 읽기 쓰기 참조가 있는지 여부를 판단하는 동시에 캐쉬 플러쉬(flush) 시점까지 해당 블록이 히스토리 버퍼에 존재하는지 여부를 판단하여 조기 방출 블록인지 여부를 판단하는 것이 바람직하다. 즉, 본 발명의 실시예에 따른 버퍼캐쉬 관리방법은 캐쉬 플러쉬(flush) 시점 전까지 해당 블록이 쓰기 참조가 1회의 단발성이고, 읽기 참조가 없고, 히스토리 버퍼에 존재하지 않는 경우에 조기 방출 블록으로 판단하게 된다.
- [0055] 쓰기 횟수가 1회여서 삭제된 블록이 다시 캐쉬에 진입한 경우 그 쓰기 횟수를 0부터 재계산할 경우 해당 블록이 캐쉬에서 삭제 후 재 진입되는 일이 반복될 수 있다. 따라서, 버퍼캐쉬에서 데이터가 삭제되더라도 해당 블록의 쓰기 발생 여부는 히스토리 버퍼(history buffer)를 두어 한동안 저장하도록 한다.
- [0056] 예를 들어 도 5에 나타낸 바와 같이, 블록 B는 5초 지점에서 읽기 참조가 없고 쓰기 횟수만 1회이므로 방출 플래그가 'o'으로 설정돼 있어 스토리지 반영(flush) 후 캐쉬에서 삭제되지만, 7초 지점에 다시 쓰기 참조가 발생하여 캐쉬에 진입할 때는 히스토리 버퍼에 블록 B가 존재하므로(즉, 최초의 쓰기가 아니므로) 방출플래그는 'x'로 바뀌게 된다.
- [0058] **성능 평가**
- [0059] 이하에서는 본 발명의 실시예에서 제안한 기법의 성능을 평가하기 위해 수행한 시뮬레이션 실험 결과를 기술한다. 버퍼캐쉬에서 데이터를 관리하는 단위인 블록 크기는 대부분의 운영체제에서 사용하는 4KB로 설정하였다. 실험에 사용된 DRAM 및 플래시메모리의 성능과 소모전력은 [표 1]과 같다.

표 1

매체	연산	접근 시간	전력(에너지)소모
DRAM	read	50(ns)	0.1(nJ/bit)
	write	50(ns)	0.1(nJ/bit)
	refresh	-	1(W/GB)
플래시 메모리	read	284.2(μs/4KB)	9.5(μJ/4KB)
	write	1833(μs/4KB)	76.1(μJ/4KB)

[0061]

[0063] DRAM의 읽기/쓰기 접근 시간은 50ns이고 읽기/쓰기 시 소모되는 전력량은 비트 당 0.1nJ이다. 한편, DRAM의 경우 읽기/쓰기가 없는 휴지상태에도 지속적인 전력재공급(refresh) 연산을 통해 정적전력(static power)이 GB 당 1W가 소모되며 이를 실험에 반영하였다. 플래시메모리의 읽기/쓰기 접근 시간은 4KB 페이지당 각각 28.4.2μs와 1833μs이고, 전력소모는 9.5μJ과 76.1μJ이다.

[0064] 실험에 사용된 트레이스는 스마트폰에서 사용되는 브라우저(Browser), 페이스북(Facebook), 팜스토리(Farmstory), No5, 네이버맵(Navermap), 틱톡(Tic toc), 월드워(Worldwar) 등 7가지 앱 실행시의 파일 참조 트레이스와 여러 앱을 섞어서 수행한 경우의 파일 참조 트레이스인 Mix1, Mix2로 구성된다. 트레이스는 각 앱을 15-20분간 실행하면서 발생한 파일 참조 기록을 추출하였다.

[0065] 실제 환경에서 스마트폰 사용자가 여러 앱을 사용하는 것과 유사한 조건에서 제안한 기법의 성능을 평가하기 위해 Mix1은 페이스북(Facebook)과 네이버맵(Navermap), No5를 함께 수행하였고, Mix 2는 브라우저(Browser)와 틱톡(Tictoc), No5, 페이스북(Facebook)을 함께 수행하였다. [표 2]는 추출된 트레이스의 특성을 정리한 것이다.

표 2

Workload	Total # of references	Ratio of read/write
Browser	26,221	1:3.38
Facebook	25,962	1:2.15
Farmstory	34,360	1:4.17
No5	155,299	1:1.93
Navermap	407,857	1:18.03
Tictoc	31,906	1:5.87
Worldwar	12,320	1:44.15
Mix1	628,739	1:6.05
Mix2	438,410	1:1.23

[0067]

[0069] 도 6 내지 도 8은 캐쉬 크기의 변화에 따라 종래 캐쉬 관리기법과 제안한 본 발명의 실시예에 따른 관리방법의 캐쉬 부재율(miss ratio) 나타낸 그래프이다. 도 6은 브라우저, 페이스북, 팜스토리 실행에 따른 캐쉬 부재율을 나타내고, 도 7은 네이버맵, No5 및 틱톡 실행에 따른 캐쉬 부재율을 나타내며, 도 8은 월드와, Mix1 및 Mix2 실행에 따른 캐쉬 부재율을 나타낸다.

[0070] 종래의 기법은 LRU(least recently used) 캐쉬 교체 알고리즘과 5초 단위의 스토리지 반영 연산(flush)을 사용하며, 제안한 본 발명의 실시예에 따른 버퍼캐쉬 관리방법은 반영 연산 시점에 1회의 쓰기만 발생한 블록을 캐쉬에서 조기 방출한다.

[0071] 도 6 내지 도 8에 나타낸 바와 같이, 모든 경우에서 제안한 본 발명의 실시예에 따른 관리방법이 종래의 기법보다 성능이 향상되었으며, 성능 향상 폭은 6-33%에 이르렀다. 특히, 제안한 본 발명의 실시예에 따른 버퍼캐쉬 관리방법의 성능 개선은 캐쉬 크기가 작을수록 더 큰 것을 확인할 수 있었다.

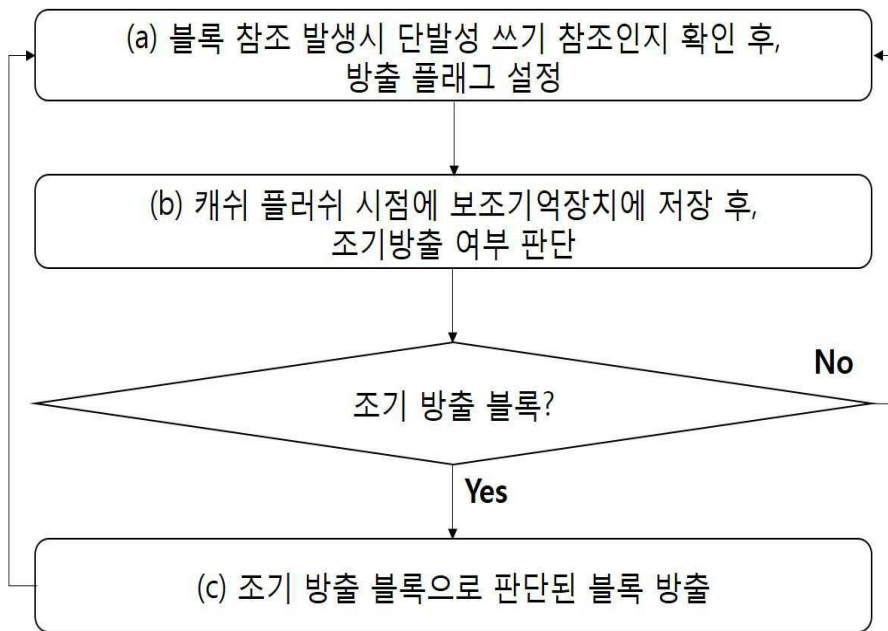
[0072] 제안한 본 발명의 실시예에 따른 버퍼캐쉬 관리방법의 캐쉬 공간 확보 효과가 어느 정도인지를 알아보기 위해

동일한 성능을 나타내는 데에 필요한 캐쉬 공간이 기존 방식과 제안한 방식에서 어떻게 다른지 비교해 보았다.

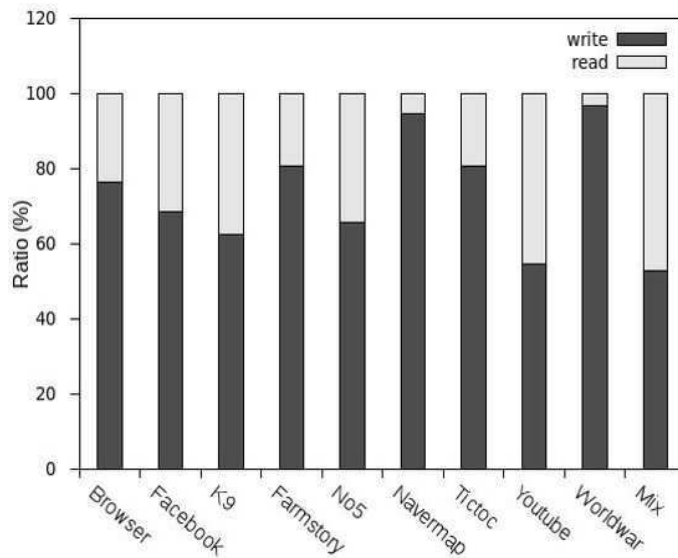
- [0073] 도 9는 Mix1 워크로드에서 동일한 성능을 나타내는 두 지점인 A와 B에서 기존 기법과 본 발명의 실시예에 따른 버퍼캐쉬 관리방법에 필요한 캐쉬 크기를 나타내는 그래프이다. 도 9에 나타난 바와 같이, 제안한 본 발명의 실시예에 따른 버퍼캐쉬 관리방법은 기존 기법에 비해 동일한 성능을 나타내기 위해 1/4 이하의 캐쉬 공간이 필요한 것을 확인할 수 있다.
- [0074] 도 10 및 도 11은 동일한 캐쉬 성능을 나타내는 지점에 대해 기존 기법과 제안한 본 발명의 실시예에 따른 버퍼캐쉬 관리방법의 총 수행시간과 전력소모를 비교한 그래프이다. 도 10 및 도 11에 나타난 바와 같이, 제안한 본 발명의 실시예에 따른 관리방법은 기존 기법에 비해 상대적으로 더 작은 캐쉬 크기를 가지기 때문에 성능은 더 나쁠 수 있으나 DRAM에 의한 전력 소모가 더 적은 것을 기대할 수 있다.
- [0075] 그러나, 도 10에 나타난 바와 같이, 제안한 본 발명의 실시예에 따른 관리방법이 비록 1/4 이하의 캐쉬 크기를 가졌음에도 기존 기법에 비해 성능이 거의 저하되지 않는 것을 확인할 수 있다. 이는 성능 개선 효과가 없는 1회성 쓰기 참조 블록을 캐쉬에서 빨리 삭제해서 추가적인 공간 확보를 함으로써 얻게 된 결과이다.
- [0076] 또한 도 11에 나타난 바와 같이, 제안한 본 발명의 실시예에 따른 관리방법은 기존 기법에 비해 에너지 소모가 크게 줄어든 것을 확인할 수 있다. 기존 기법 대비 에너지 절감률은 27-92%에 이르는 것으로 확인되었다. 이는 스마트폰 전력 소모의 상당 부분을 DRAM 메모리가 차지하고 있다는 기존 연구와도 일관성이 있다.
- [0077] 스마트폰은 대부분의 시간을 휴지 상태(idle state)로 있지만, 사용자가 스마트폰 사용을 원할 때는 언제든지 재개(resume)할 수 있도록 항상 대기 상태에 있어야 한다. 이를 위해 DRAM 메모리는 읽기 쓰기 연산이 없더라도 지속적인 전력 재공급(refresh) 연산을 해주어야 하며 이는 막대한 배터리 소모의 원인이 되고 있다. 제안한 본 발명의 실시예에 따른 버퍼캐쉬 관리방법은 DRAM으로 구성되는 버퍼캐쉬의 크기를 줄여 전체 시스템의 전력소모를 줄일 수 있음을 보인 것이다.
- [0078] 이와 같이 본 발명에서는 스마트폰의 파일 참조 중 상당 부분이 1회성 쓰기 참조라는 것을 분석하고 이러한 특성을 반영하는 새로운 버퍼캐쉬 관리방법을 제안한다. 제안한 본 발명의 실시예에 따른 버퍼캐쉬 관리방법은 쓰기 참조만 1번 발생한 블록을 스토리지 반영 시점에 캐쉬에서 조기 방출함으로써, 캐쉬 공간의 불필요한 낭비를 막는다. 그리고, 다양한 스마트폰 앱을 이용한 실험 결과 제안하는 본 발명의 실시예에 따른 버퍼캐쉬 관리방법은 스마트폰 성능을 5-33% 향상시킴을 확인하였고, 전력소모를 평균 64% 줄일 수 있음을 확인하였다.
- [0080] 본 명세서에서 설명되는 실시예와 첨부된 도면은 본 발명에 포함되는 기술적 사상의 일부를 예시적으로 설명하는 것에 불과하다. 따라서, 본 명세서에 개시된 실시예들은 본 발명의 기술적 사상을 한정하기 위한 것이 아니라 설명하기 위한 것이므로, 이러한 실시예에 의하여 본 발명의 기술 사상의 범위가 한정되는 것은 아님은 자명하다. 본 발명의 명세서 및 도면에 포함된 기술적 사상의 범위 내에서 당업자가 용이하게 유추할 수 있는 변형예와 구체적인 실시 예는 모두 본 발명의 권리범위에 포함되는 것으로 해석되어야 할 것이다.

도면

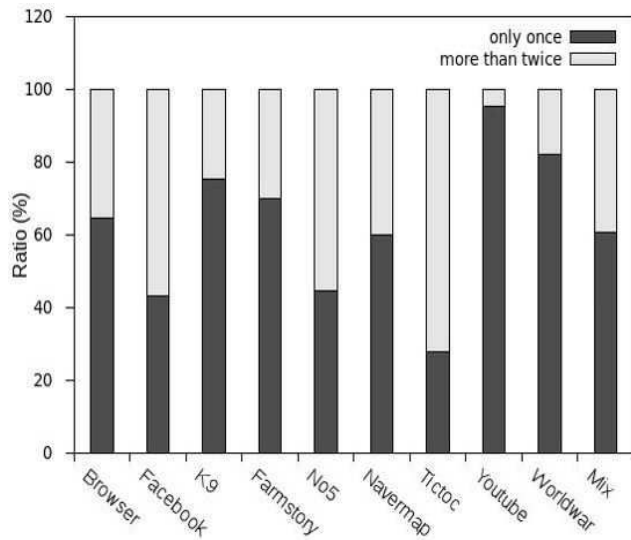
도면1



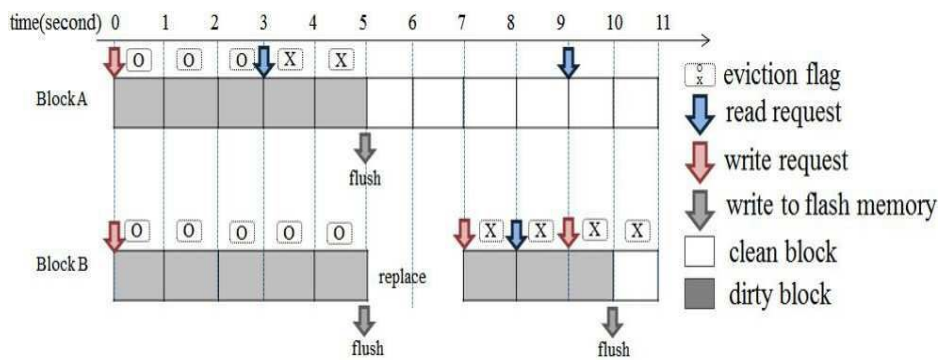
도면2



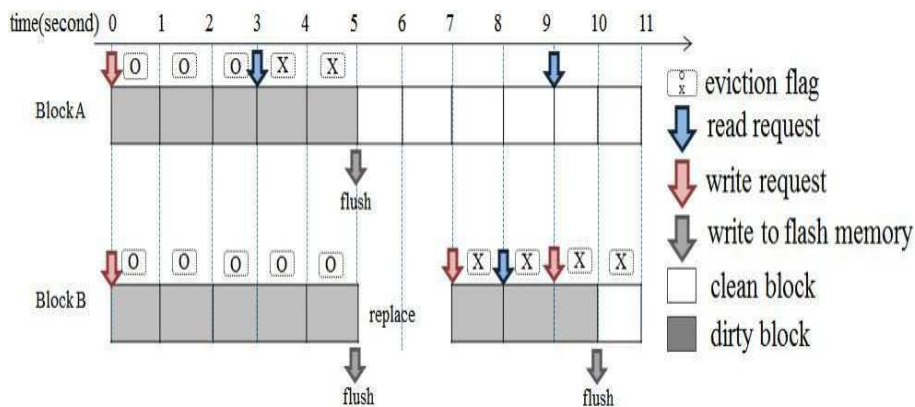
도면3



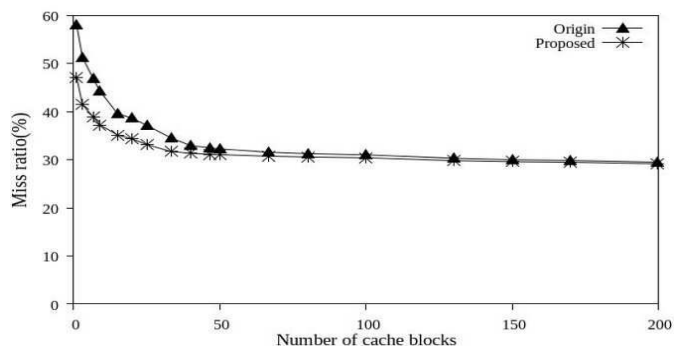
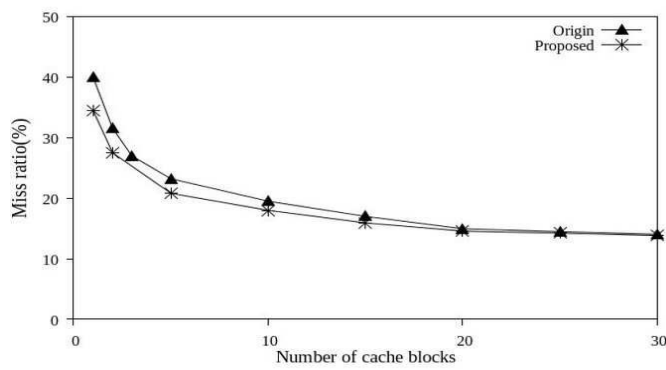
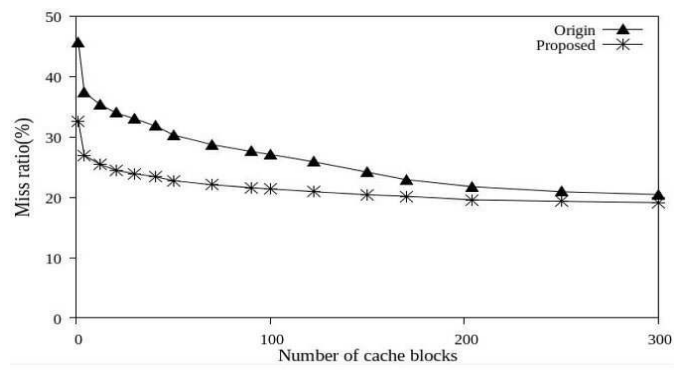
도면4



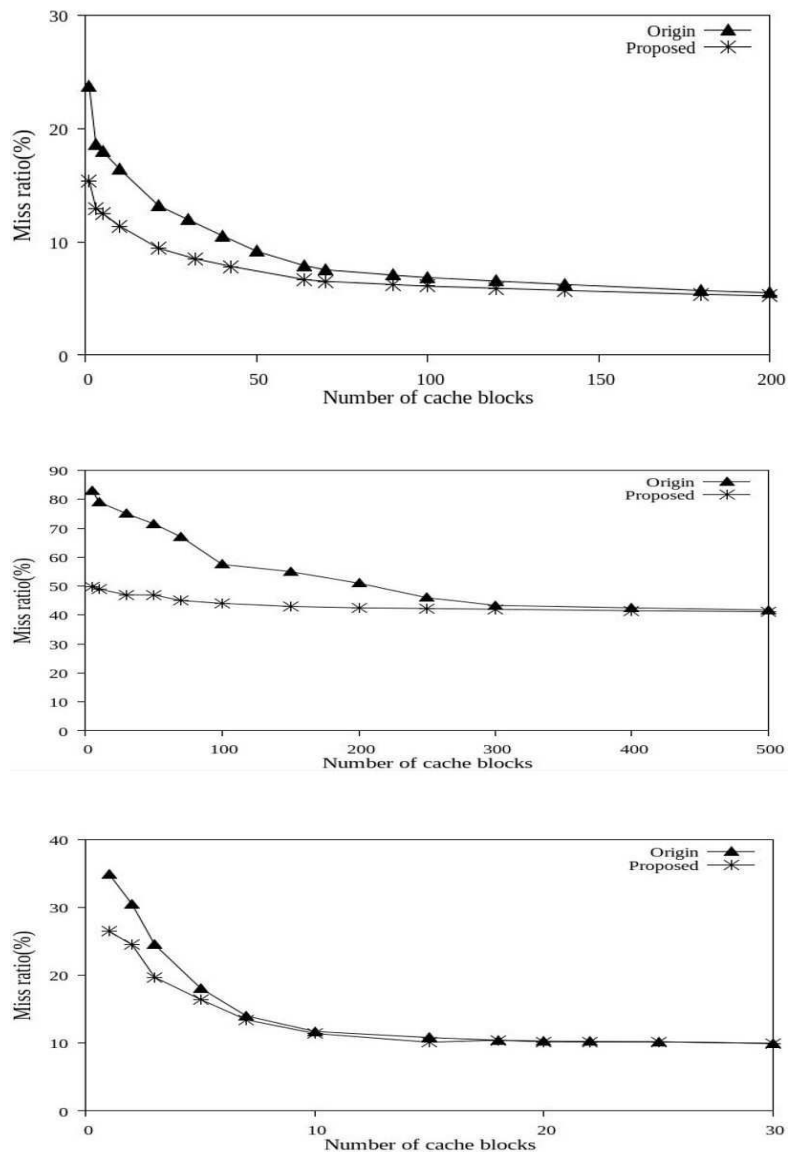
도면5



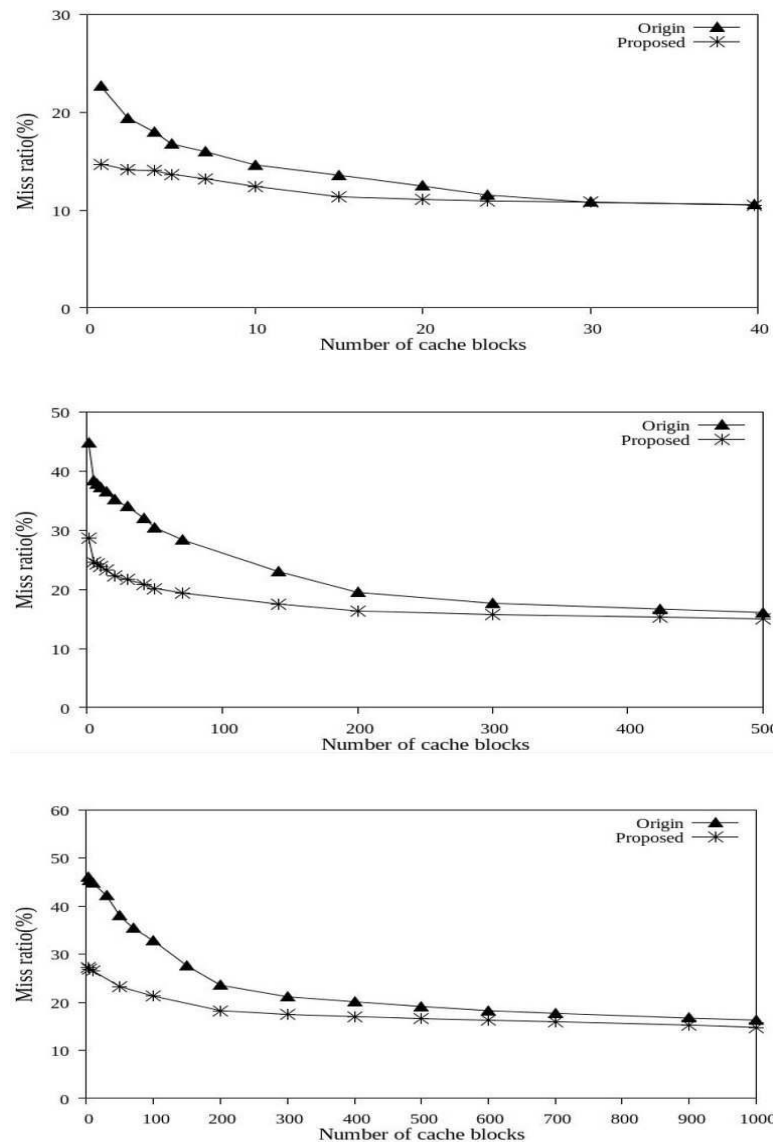
도면6



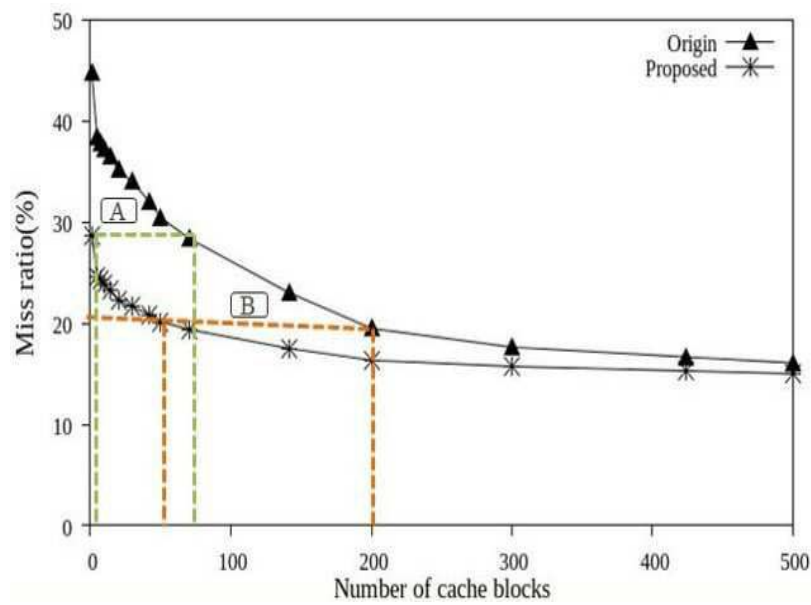
도면7



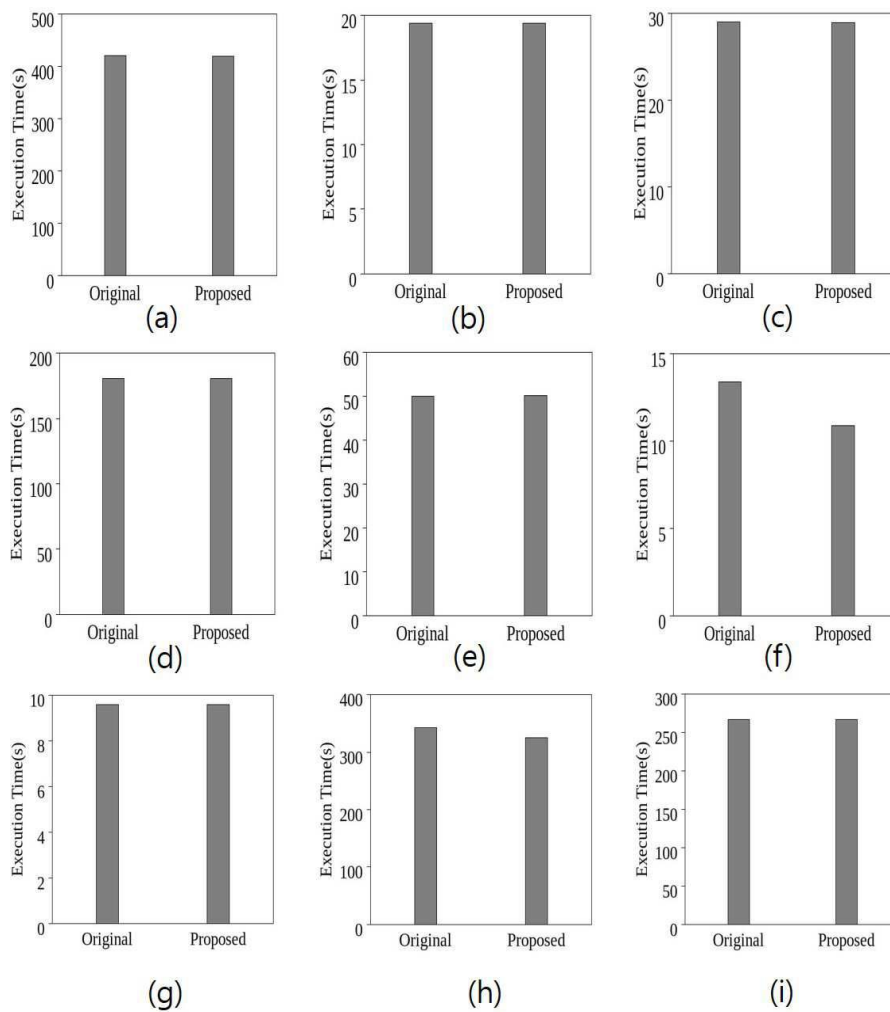
도면8



도면9



도면10



도면11

