논문 2013-50-2-17

SSD 기반 서버급 스토리지를 위한 지역성 기반 청킹 정책을 이용한 데이터 중복 제거 기법

(Data Deduplication Method using Locality-based Chunking policy for SSD-based Server Storages)

이 승 규*, 김 주 경*, 김 덕 환**

(Seung-Kyu Lee, Ju-Kyeong Kim, and Deok-Hwan Kim)

요 약

낸드 플래시 기반 SSD(Solid State Drive)는 빠른 입출력 성능, 저전력 등의 장점을 가지고 있어, 타블릿, 데스크탑 PC, 스마트폰, 서버 등의 저장장치로 널리 사용되고 있다. 하지만 SSD는 쓰기 횟수에 따라서 마모도가 증가하는 단점이 있다. SSD의 수명을 향상시키기 위해 다양한 데이터 중복제거 기법이 도입되었으나, 일반적인 고정 크기 분할방식은 데이터의 지역성을 고려하지 않고 청크크기를 할당함으로써, 불필요한 청킹 및 해시값 생성을 수행하는 문제점이 있으며, 가변 크기 분할방식은 중복제거를 위해 바이트 단위로 비교하여 과도한 연산량을 유발한다. 본 논문에서는 SSD 기반 서버급 스토리지에서 쓰기 요청된 데이터의 지역성에 기반한 적응형 청킹 정책을 제안한다. 제안한 방법은 중복데이터가 가지는 응용프로그램 및 파일 이름 기반 지역성에 따라 청크 크기를 4KB 또는 64KB로 적응적으로 분할하여, 청킹 및 해시값 생성에 따른 오버헤드를 감소시키고, 중복 쓰기를 방지한다. 실험결과, 제안하는 기법이 기존의 가변 크기 분할 및 4KB의 고정 크기 분할을 이용한 중복제거 기법보다 SSD의 쓰기 성능이 향상되고 전력 소모 및 연산시간을 감소시킬 수 있음을 보여준다.

Abstract

NAND flash-based SSDs (Solid State Drive) have advantages of fast input/output performance and low power consumption so that they could be widely used as storages on tablet, desktop PC, smart-phone, and server. But, SSD has the disadvantage of wear-leveling due to increase of the number of writes. In order to improve the lifespan of the SSD, a variety of data deduplication techniques have been introduced. General fixed-size splitting method allocates fixed size of chunk without considering locality of data so that it may execute unnecessary chunking and hash key generation, and variable-size splitting method occurs excessive operation since it compares data byte-by-byte for deduplication. This paper proposes adaptive chunking method based on application locality and file name locality of written data in SSD-based server storage. The proposed method split data into 4KB or 64KB chunks adaptively according to application locality and file name locality of duplicated data so that it can reduce the overhead of chunking and hash key generation and prevent duplicated data writing. The experimental results show that the proposed method can enhance write performance, reduce power consumption and operation time compared to existing variable-size splitting method and fixed size splitting method using 4KB.

Keywords : SSD, Data Deduplication, Locality based Chunking, Fixed-size Splitting, Variable-size Splitting

* 학생회원, ** 정회원, 인하대학교 전자공학과
(Department of Electronic Engineering, Inha University)
※ 이 논문은 2012년도 정부(교육과학기술부)의 재원으로 한국연구재단의 기초연구사업 지원을 받아 수행된 것임 (2012-0001773)
※ 본 과제(연구)는 지식경제부와 한국산업기술진흥원의 전략기술 인력양성 사업으로 수행된 결과임
※ 본 논문은 인하대학교 연구비 지원에 의하여 연구되었음
접수일자: 2012년10월19일, 수정완료일: 2013년1월18일

Ⅰ. 서 론

클라우드 컴퓨팅 및 모바일 통신 서비스의 사용량이 급격히 증가함에 따라 데이터의 처리량도 증가하고 있다. 2011년 IDC에서 발표한 보고서에 따르면 전세계에서 생성 및 복제되는 데이터의 용량은 약 2052 엑사바이트(Exabyte)에 이를 것이라고 예측을 하고 있다[1]. 최근 데이터를 저장하는 스토리지 장치로는 기계적 방식의 HDD(Hard Disk)에 비해, 전자기적으로 동작하며 소비 전력이 작고 우수한 데이터 접근 성능을 보이는 SSD(Solid State Disk)가 이슈화 되고 있다. 이와 같은 이점 때문에 모바일 기기, 데스크탑 PC등에서 SSD를 사용하고 있으며 고성능 서버와 데이터 센터 등에서도 SSD 기반 서버급 스토리지의 사용이 증가하고 있다[2]. 하지만 NAND 플래시메모리 기반의 SSD는 마모도 특성을 가지고 있어, 대용량 데이터를 처리하는 서버급 저장장치에서 효율적인 데이터 처리 및 공간 확보가 중요하다. 이를 보완하기 위해 다양한 데이터 중복제거 기술이 제안되고 있으며[3], 이 기술은 일반적으로 해시 함수를 이용하여 파일의 지문(Finger Print)을 생성하고 생성된 지문들을 비교하여 중복 유무를 판단함으로써 지문이 동일한 데이터는 제거하고 고유한 내용만을 저장함으로써 쓰기횟수 및 전력소모를 감소시킬 수 있다.

중복 데이터는 동일한 애플리케이션에 의해 데이터가 생성될 때 중복데이터의 비율이 증가하는 응용프로그램 기반 지역성과 데이터 복사시 비슷하거나 같은 파일이름으로 저장하는 파일이름 기반 지역성 특성이 있다[4].

데이터 중복제거 기술은 전송하고자 하는 데이터를 고정 크기 분할기법(FSC:Fixed Size Chunking)과 가변 크기 분할기법(VSC: Variable Size Chunking)을 적용하여 데이터 중복제거 알고리즘을 수행하는 2가지로 분류할 수 있다. 이 기법들 중 고정 크기 분할 기법은 청크 사이즈를 너무 작게 설정하여 데이터 중복제거 알고리즘을 적용했을 경우 각 청크들의 해시값 계산에 따른 CPU 및 물리적 자원들을 많이 소모하여 시스템의 오버헤드를 증가 시켜 입출력 성능이 우수한 SSD의 장점을 최대한 활용할 수 없다. 반면에 청크 사이즈를 너무 크게 설정하여 데이터 중복제거 알고리즘을 적용했을 경우 해시 값 계산양은 줄일 수 있으나, 중복 제거율은 낮아지는 문제점이 있다.

본 논문은 중복 데이터의 응용프로그램 기반 및 유사한 파일 이름을 가지는 지역성의 유사비율에 따라 청크 크기 분할을 달리 하는 지역성 기반 청킹 정책을 제안한다. 이 기법은 서버로 전송되는 데이터의 메타데이터를 이용하여 지역성 특성의 일치여부 및 유사비율을 검사하고, 유사비율에 따라 데이터 중복제거 알고리즘 수행 전에 청크 크기를 적응적으로 할당함으로써 시스템의 성능 향상 및 전력 소모 감소 효과를 기대할 수 있다.

본 논문의 구성은 다음과 같다. Ⅱ장에서 본 연구와 관련된 연구 내용에 대해 기술하고, Ⅲ장에서 제안하는 지역성 기반 청킹 기법을 이용한 데이터 중복제거 기법에 대해 설명한다. Ⅳ장에서 실험 및 성능 평가 결과에 대해 기술하였으며, 마지막으로 Ⅴ장에서 결론 및 향후 연구방향을 제시한다.

Ⅱ. 관 련 연 구

2.1 데이터 중복제거 기술의 분류

데이터 중복제거 기술은 중복제거를 수행하는 방식에 따라 즉시 중복제거(In-line) 방식과 예약 중복제거(Post-process) 방식으로 분류할 수 있다[5]. 즉시 중복제거 방식은 데이터가 타깃 스토리지에 저장되기 전에 중복제거를 수행하는 기법으로, 마모도 특성을 가지는 SSD의 쓰기 연산을 줄일 수 있는 장점이 있지만, 연산량이 많을 경우 중복제거 알고리즘에 따른 병목 현상이 발생하는 단점이 있다. 예약 중복제거 방식은 네이티브 백업 이미지를 저장하기 위한 스토리지가 추가로 필요한 특징이 있어 추가 비용이 발생하는 단점이 있지만, 네이티브 백업 이미지를 통한 디스크 타겟에서 신속한 복구가 가능한 장점이 있다. 타깃 스토리지로 전송되는

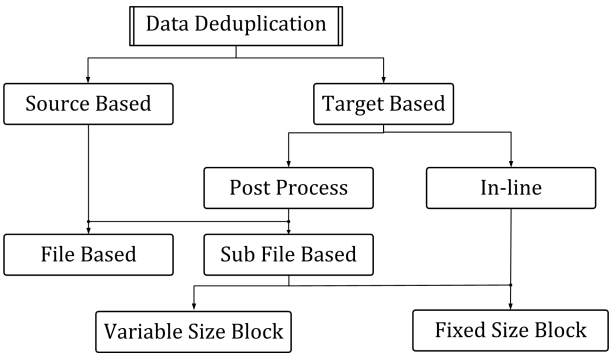

그림 1. 데이터 중복제거 기술의 분류
Fig. 1. Classification of data deduplication technology.

데이터는 중복제거를 위해 고정 크기 분할기법과 가변 크기 분할기법으로 분류하여 사용할 수 있다. 그림 1은 데이터 중복제거 기술의 분류를 나타낸다.

고정 크기 분할기법을 적용한 중복제거 기법은 전송되는 데이터를 같은 크기의 청크로 나누고 각각 청크들에 해시 값을 부여한다. 매번 새로운 청크들이 해시 될 때마다, 해시 테이블에 저장되어 있는 해시 값들과 비교되며, 만약 비교 대상 해시 값이 존재하면 해당 청크는 중복된 데이터이므로 저장하지 않고, 그 해시 값은 해당 핑거프린트에 추가되며, 이와 같은 연산을 계속 반복한다. 이 기법은 해당 데이터가 수정될 경우 각 청크의 해시 값이 틀려지므로 중복제거를 수행할 수 없는 문제점이 발생할 수 있지만 연산속도가 빠르다.

가변 크기 분할기법을 적용한 중복 제거 기법은 고정 청크 단위처럼 해시 값을 비교하여 중복여부를 판단할 수 있다. 이 기법은 파일에 BSW(Basic Sliding Window) 방식[6]을 적용해 해당 데이터의 해시 값을 미리 정해 놓은 비트 패턴과 일치하는지 비교하고 일치하지 않으면 1Byte씩 이동하며 일치하는 해시 값을 찾는다. 만약 해시값이 일치하면 , 그 지점을 경계로 청킹한다. 해시 값을 비교하기 위한 자원소모량이 많기 때문에 4KB의 최솟값과 64KB의 최댓값을 지정하여 사용하고 있다[7]. 이 기법은 고청 청크 단위보다 연산시간이 느리지만 중복 제거율은 우수하다.

2.2 선행된 데이터 중복제거 기술

중복 데이터를 효과적으로 제거하기 위한 대표적인 연구로는 Venti[8], LBFS(Low-Bandwidth Network File System)[9] 등이 있다. LBFS는 네트워크 환경이 좋지 않은 환경을 고려하여 설계된 네트워크 파일 시스템으로써, 네트워크 연결이 잠시 중단된 상태에서 다시 연결을 하더라도 데이터 전송이 가능하도록 프로토콜을 설계 하였고 한정된 네트워크 대역폭을 효과적으로 사용하기 위해 중복 데이터의 재전송을 막는 기술을 고안하였다. 이 기술은 가변 크기 분할 기법이라고도 불리는 CDC(Content-Define Chunks) 방식[6]을 사용하며, Rabin Fingerprint[10]로 해시하여 임의의 경계 값을 설정하여 데이터 전송 전에 해당 청크를 MD5, SHA-1[11]과 같은 해시함수를 이용하여 해시값를 만들어 중복제거를 실행한다. Venti는 데이터를 고정된 크기의 블록으로 청킹하고 각 블록에 SHA-1으로 해싱하여 중복여부를 식별하는 스토리지 시스템이다. Venti는 고정크기 블록을 사용해 byte by byte로 중복여부를 비교하는 가변 크기 분할기법 보다 오버헤드가 적어 빠른 성능을 보이나, 파일을 수정하거나 데이터가 변경된 경우 중복 블록들의 오프셋 위치가 달라짐에 따라 수정이 일어난 뒷부분의 블록들은 다른 해시 값으로 계산하게 되어 중복을 찾아낼 수 없는 데이터 쉬프팅(shifting) 문제가 발생한다. Data Domain, Inc.와 프린스톤 대학교에서는 중복 제거 파일 시스템에서의 디스크 병목 현상을 피하기 위해 디스크 엑세스를 최소화하고 데이터 중복제거를 하기위한 서버급 스토리지의 관한 기술을 연구 중이다[12]. SHA-1 기반 중복제거 기법인 ChunkStash[13]는 각 청크의 메타데이터를 NAND 플래시 메모리에 캐싱하여 중복제거의 연산시간을 감소시켰다.

데이터 중복제거 기법을 SSD에 적용한 대표적인 연구로는 Gupta의 CA-SSD[14], Chen의 CA-FTL[15]과 SSD에 중복제거 프레임워크를 제시한 Chin-Hsien Wu의 연구[4]가 있다. CA-SSD는 Value-locality를 기반으로 낸드 플래시에 최적화된 주소 사상 기술을 제안하였다. CA-FTL은 데이터 중복제거를 통한 SSD의 마모도를 줄이기 위해 고안된 방법으로써 해싱 오버헤드를 줄이기 위해 Pre-Hashing, Sampling, dynamic switch 등의 기법 및 2단 레벨 매핑 테이블을 구현하여 N:1 매핑이 가능하게 함으로써 시스템의 성능을 높였으며, 즉시 중복제거 방식(In-line deduplication)에 최적화한 기법이다. 즉시 중복제거 방식은 전송되는 데이터량이 많을 경우 많은 CPU 자원 소모 때문에 데이터 병목현상이 일어 날 수 있다. 본 연구에서 제안한 방법은 데이터에 청킹알고리즘을 적용하기 전에 메타데이터 정보를 가지고 중복데이터의 특성인 지역성을 고려하여 청크 크기를 할당함으로써 SSD의 쓰기 성능을 최대한 활용할 수 있다.

Ⅲ. 지역성 기반 청킹 정책을 이용한 데이터 중복제거 기법

3.1 제안하는 지역성 기반 청킹 정책을 이용한 데이터 중복제거 기법

SSD 기반 서버급 스토리지 시스템에서 즉시 중복제거 방식을 사용시 전송되는 데이터의 청킹 및 해시값 생성에 따른 CPU 과부하에 의해서 디스크 병목현상이

발생할 수 있다. 본 논문에서는 중복 데이터의 지역성 특성에 기반한 적응형 청킹 정책을 적용함으로써 CPU 과부하를 줄여 저전력 및 SSD의 쓰기 속도를 최대 활용할 수 있다. 또한, 기존의 가변 크기 분할기법과 고정 크기 분할기법은 데이터의 지역성에 상관없이 중복제거가 수행되지만 제안하는 지역성 기반 청킹 정책은 데이터가 서버에 전송되는 도중 1차적으로 메타데이터 비교를 통해 청크 크기를 할당하고 2차적으로 중복제거를 수행하므로 차별성이 있다.

SSD는 기존 하드디스크의 탐색시간 과부하를 없앨 수 있는 장점과 내부 설계시 병렬적 전송을 통한 우수한 데이터 전송 속도를 지닌 저장장치이다. 이러한 SSD를 서버급 스토리지로 효과적으로 활용하기 위해 본 논문에서 제안하는 지역성 기반 청킹 정책이 적용된 데이터 중복제거 기법의 파일 시스템 구조는 그림 2와 같다. 제안한 방법은 Dedup 파일 시스템 상위 단에 Meta Data Check Layer를 추가하여 전송되는 데이터의 지역성 특성의 일치여부 및 유사비율에 따라 청크 크기를 고정된 4KB로 사용할지 아니면 고정된 64KB로 사용할지를 판단하여 실행한다. 이 기법은 byte by byte로 해시값의 중복여부를 판단하는 가변 크기 분할 기법보다 중복 제거율은 낮을 수 있으나 쓰기성능은 향상시킬 수 있다.

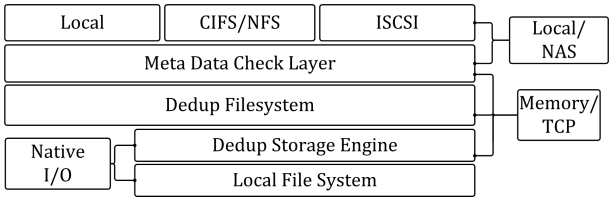

그림 2. 지역성 기반 청킹 정책이 적용된 데이터 중복 제거 기법의 파일 시스템 구조

Fig. 2. File system structure of data deduplication method that applying locality based chunking policy.

표 1. 각 메타 테이블의 구성요소

Table 1. Components of each meta table.

List	Value
파일 이름 아스키코드	3248493295119....
파일 이름	2012_work
파일 확장자	ppt
파일 확장자 아스키코드	707074
파일 크기	342123 Byte
수정된 시간	2012.07.05 21.08.07

메타 데이터 체크 레이어 단에는 서버 스토리지에 저장되어 있는 데이터들의 메타데이터들이 램에 저장되어진다. 메타데이터는 파일의 이름과 확장자, 파일크기, 파일이 수정된 시간, 파일 이름과 확장자의 아스키코드 값 등을 포함한다. 이를 각 파일 확장자별로 메타데이터 테이블들을 구성하였다. 메타테이블을 구성하는 데이터는 SSD의 spare 영역에 저장되어 있는 메타데이터들을 사용한다. 표 1은 각 메타데이터 테이블의 예를 보여준다.

3.2 제안하는 지역성 기반 청킹 정책을 이용한 데이터 중복제거 기법의 알고리즘 및 시스템 구조

중복 데이터의 응용프로그램 기반 지역성과 파일 이름 기반 지역성 특성을 이용한 데이터의 중복여부를 유추하기 위해 메타데이터의 파일 확장자 아스키코드와 파일 이름의 아스키코드를 참조한다. 메타데이터의 파일 확장자 아스키코드는 동일한 응용프로그램을 이용해 생성된 데이터인지를 검사하고, 파일 이름 아스키코드는 파일 확장자 아스키코드의 일치 여부에 따라 비교를 수행한다.

그림 3은 파일 확장자 아스키코드의 일치 여부와 파일 이름 아스키코드에 대한 유사비율을 비교하는 과정을 보여준다. 파일 확장자 아스키코드의 일치여부와 파일 이름 아스키코드의 유사비율은 직접적인 아스키 코드값들 간의 비교를 통해 구할 수 있다. 쓰기요청된 데이터의 메타데이터로부터 아스키 코드 값을 얻고, 메타

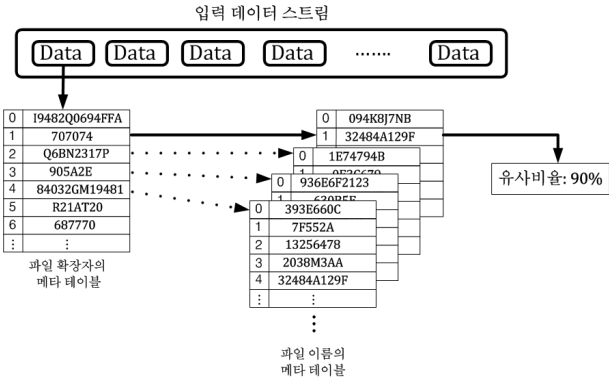

그림 3. 파일 확장자의 일치 여부와 파일 이름에 대한 유사비율의 검사과정

Fig. 3. Inspection process of similarity ratio between file names and file extensions.

테이블에 저장되어 있는 파일 확장자의 아스키코드 값과의 일치여부를 판단 후 일치할 경우 파일 이름들의 아스키 코드값들과 유사비율을 검사한다. 그림 3의 경우는 메타테이블에 저장되어 있는 파일 확장자의 아스키 코드들을 엔트리 별로 검색한 결과 1번 엔트리와 일치함을 확인하고, 이 후 해당 엔트리 확장자들의 파일 이름 아스키 코드들을 엔트리별로 검색하여 3번 엔트리의 유사비율이 90%로 매칭되는 결과를 보여준다.

본 연구에서는 중복데이터의 두 가지 특성에 기반해 청크 크기를 적응적으로 할당하기 위해 메타 데이터의 파일 확장자 아스키코드 일치 여부와 파일 이름 아스키 코드의 유사비율에 따라 4KB 또는 64KB의 크기로 청킹한다. 쓰기 요청되는 데이터의 지역성 특성을 이용해 파일 확장자와 이름의 일치 여부 및 유사비율을 검사함

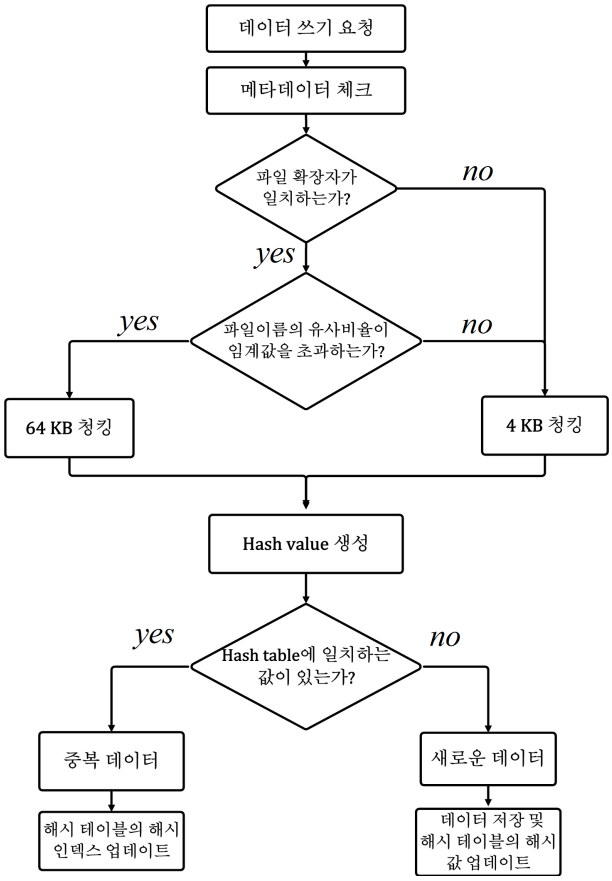

그림 4. 파일이름 기반 지역성 및 응용프로그램 기반 지역성에 따른 청킹 정책을 이용하는 데이터 중복제거 알고리즘

Fig. 4. Data deduplication algorithm using chunking policies based on file name locality and application based locality.

으로써 중복여부를 유추하고, 적응적으로 청킹함으로써 청킹 오버헤드와 청크에 대한 해시값 생성에 따른 CPU 부하를 효과적으로 줄일 수 있다. 또한, SSD의 쓰기 속도를 향상 시키고, 전력 소모를 감소 시킬 수 있다. 그림 4는 파일이름 기반 지역성 및 응용프로그램 기반 지역성에 따른 청킹 정책을 이용하는 데이터 중복제거 알고리즘이다.

먼저 데이터 쓰기 요청이 들어오면 전송되는 메타 데이터를 후킹하여 메타테이블 영역에 저장된 파일 확장자들의 아스키코드 값의 일치여부를 비교한다. 만약 파일 확장자의 아스키코드 값이 불일치할 경우 4KB의 고정 크기 분할 기법을 적용한다.

파일 확장자가 일치할 경우는 파일 이름의 아스키코드 값과 메타테이블에 저장되어 있는 아스키코드 값들과의 유사비율을 검사한다. 만약 유사비율이 정해놓은 임계값보다 클 경우 고정 크기 분할기법을 이용하여 64KB를 할당하고, 임계값보다 작을 경우 4KB를 할당한다. 이 후, 분할된 청크에 해시값을 생성하고 해시테이블에 있는 해시값과 중복여부를 판단 후 중복데이터는 해시 테이블의 해시 인덱스만을 업데이트한다. 중복되지 않은 데이터는 스토리지에 데이터를 저장한 후, 해시테이블에 새로운 청크의 해시 값을 업데이트 한다.

파일 이름의 유사비율에 따른 임계값은 4.2절의 실험 결과를 통해 알아본다.

본 논문에서 제안하는 지역성 기반 청킹을 이용하는 시스템의 구조는 그림 5와 같다. 커널 단에서 Dedup

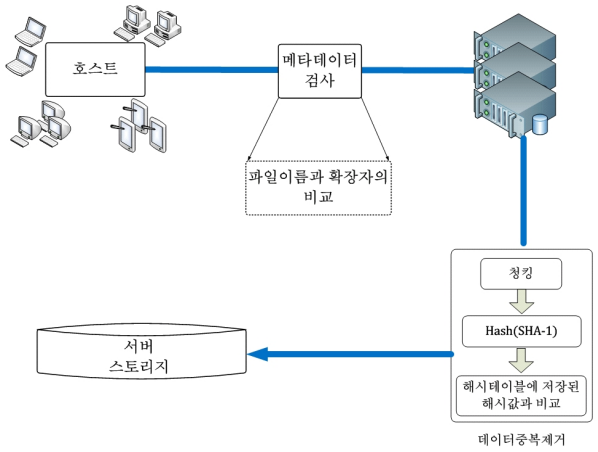

그림 5. 지역성 기반 청킹 정책을 이용하는 시스템 구조

Fig. 5. System architecture using locality-based chunking policy.

File System으로 쓰기 요청이 발생했을 때 Meta Data Check Layer를 추가하여 전송되는 데이터의 메타데이터를 후킹(Hooking)한다. 후킹한 메타데이터 정보의 응용프로그램 기반 지역성의 일치 여부와 파일 이름 기반 지역성의 유사비율을 비교하여 일정 유사비율 이상일 경우를 임계값으로 주어 적응적으로 청크크기를 할당하는 정책을 실시한다. 그 후 SHA-1 해시 함수를 이용하여 핑거프린트를 생성 후 해시테이블에 있는 핑거프린트들과 중복여부를 비교한 중복되지 않은 데이터만 스토리지에 저장한다.

Ⅳ. 실험 및 성능 평가

4.1 실험 환경

본 논문의 실험 환경은 Intel i3-2100 3.10GHz의 CPU와 4GB RAM을 사용하였다. 운영체제로는 Linux 2.6.17 환경에서 오픈 소스인 OpenDedup을 사용하고 I/O METER로 벤치마크하여 지역성 기반 청킹정책을 이용한 SSD의 쓰기 속도 및 중복율을 측정하였다. 실험에 사용되는 서버급 저장장치는 리눅스 환경에서 SSD의 특성을 가지는 64GB의 디스크 4개를 설정하여 RAID5로 묶어 가상화 서버 스토리지 환경을 구축했다. 표 2는 사용한 가상 SSD의 환경변수이다.

표 2. 가상 SSD의 환경변수
Table 2. Environmental parameters of virtual SSD.

파라미터	값
Page size	4 KB
Block size	256 KB
Page read time	25 us
Page write time	200 us
Block erase time	1.5 ms

4.2 실험 결과

본 연구에서 제안하는 방법의 성능평가를 위해 연구실내의 10여명이 공용으로 사용하는 데이터 백업 서버의 저장장치에 저장된 영상 데이터, 문서 데이터 및 각종 프로그램 데이터들을 무작위로 일별로 추출하여 벤치마크 워크로드를 구성하였다.

실험에서 사용하는 중복율과 중복 제거율의 수식은 아래의 (1)과 (2)와 같다.

$$\text{중복율} = \frac{\text{중복된 데이터}}{\text{백업스토리지에 저장된 데이터}} \times 100(\%) \quad (1)$$

$$\text{중복제거율} = 1 - \frac{\text{총쓰기횟수}}{\text{총쓰기요청횟수}} \times 100(\%) \quad (2)$$

그림 6은 각 워크로드별로 프로그램을 설치하였을 때의 청크 크기에 따른 중복율의 측정 결과이다.

각 워크로드의 중복율은 청크의 크기가 128KB일 때 최소 4.9%의 중복율을 보였고 청크의 크기가 가장 작은 4KB일 때 최대 49.5%의 중복율을 보였다. 오피스와 CAD 같은 응용 프로그램은 파일의 일부분을 수정할 때 임시 파일 및 백업 파일등이 생성 되므로 다른 워크로드들에 비해 상대적으로 높은 중복율을 보이는 특성을 볼 수 있다.

표 3의 실험 결과는 통계적으로 백업스토리지에 전

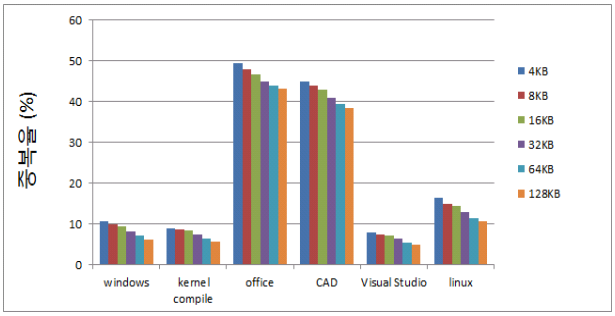

그림 6. 청크 크기에 따른 워크로드별 중복율
Fig. 6. Duplication ratio of workloads with respect to chunk size.

표 3. 파일 이름 아스키코드의 유사 비율과 청크 크기에 따른 중복 제거율
Table 3. Deduplication ratio according to similarity ratio of file name and chunk size.

② \ ①	4K	8K	16K	32K	64K	128K
100~80	84.2	82.8	81.6	80.7	80	74.5
79~60	57.2	54.9	52.7	51	50	41.8
59~40	30.2	27.4	24.7	22.8	21.1	15.3
39~20	12.2	12.2	8.6	7.7	7.5	5.2
19~0	3.3	2.4	1.9	1.7	1.5	0.8

① - 청크크기, ② - 파일 이름 아스키코드의 유사비율

송되는 데이터의 파일 이름 기반 지역성의 유사 비율과 청크 크기에 따라서 중복 제거율의 차이가 있음을 나타낸다.

백업 서버에 저장된 데이터를 10GB씩 10회 전송하여 총 100GB에 해당하는 파일 이름의 유사 비율과 청크 크기에 따라 중복제거율을 측정하고, 총 평균의 값을 표에 나타냈다. 100~80%의 유사비율을 보일 때가 가장 중복제거율이 높게 나타났다. 청크크기에 따른 중복제거율은 4KB를 사용할 때 중복제거율이 가장 우수했으며, 128KB를 사용할 때 중복제거율이 가장 낮은 결과를 보였다. 또한, 64KB와 128KB의 청크크기 간의 중복제거율 비율의 차이가 크게 나타났으며, 중복제거의 성능 또한 낮아지는 것을 확인할 수 있었다.

위의 표3의 값의 통계적 의미를 조사하기 위하여 10회의 데이터 전송에 대한 분산분석으로 검증하였으며, ($F(5,28) = 44.6$, $p<0.01$)의 결과로 의미있는 중복 제거율 성능을 보인다.

실험은 중복 제거율이 가장 높은 100~80%의 유사비율을 임계값으로 설정하고, 최소 청크 크기인 4KB와 청크크기 간의 중복제거율 비율의 차이가 가장 큰 폭을 보이는 64KB를 할당하여 제안하는 지역성 기반 청킹 정책을 이용하여 데이터 중복제거를 수행하였다.

그림 7은 제안한 기법과 기존의 4KB 및 64KB 고정 크기분할 기법과 가변 크기분할 기법을 적용한 중복 제거율을 보여준다.

제안한 방법은 기존의 4KB 고정 크기 분할기법에 비해 3.2%, 가변 크기 분할기법에 비해 5.9% 작은 중복 제거율을 보였으나 64KB 고정 크기 분할기법보다는 6% 향상된 성능을 보였다.

그림 8은 제안한 기법과 기존의 기법들과의 쓰기 성능 비교를 보여준다. 제안한 기법은 4KB의 고정 크기 분할기법에 비해 제안한 지역성 기반에 따른 청킹 정책의 쓰기 성능이 23.5% 향상됨을 보였다.

가변 크기 분할기법에 비해서도 48%의 쓰기 성능 향상을 보였으며, 반면 64KB의 고정 크기 분할기법에 비해서는 8%의 쓰기 성능 감소를 보였다.

위의 실험 결과들을 보면 제안하는 응용프로그램 지역성 및 파일 이름 기반 지역성을 가지는 중복데이터의 특성에 따라 적응적으로 청킹 정책을 수행하는 이 기법은 기존의 기법들과의 중복 제거율 차이는 작았으며,

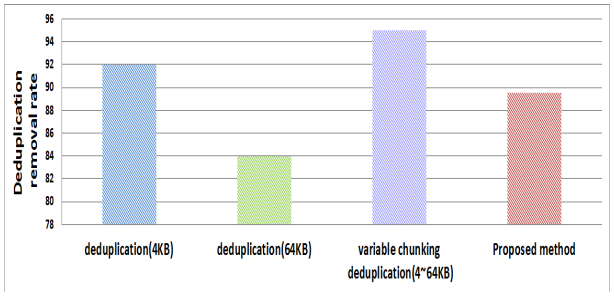

그림 7. 기존의 기법들과 제안한 기법의 중복 제거율 비교

Fig. 7. Duplication removal ratio of existing techniques and the proposed method.

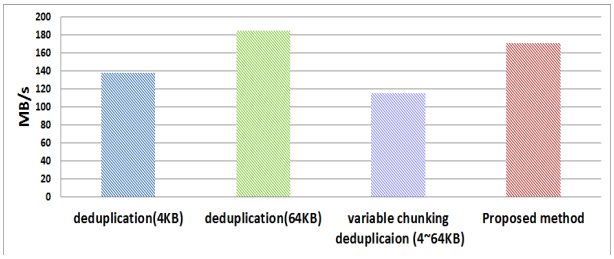

그림 8. 기존의 기법들과 제안한 기법의 쓰기 성능 비교

Fig. 8. Write performance between existing techniques and the proposed method.

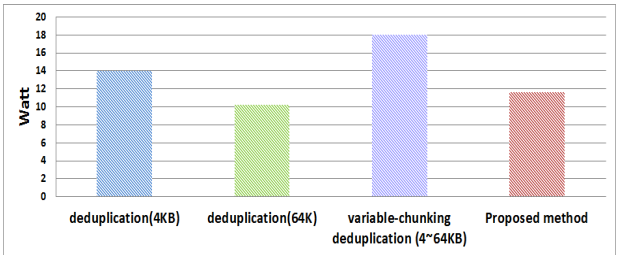

그림 9. 기존의 기법들과 제안한 기법의 전력 소모 성능 비교

Fig. 9. Power consumption between existing techniques and the proposed method.

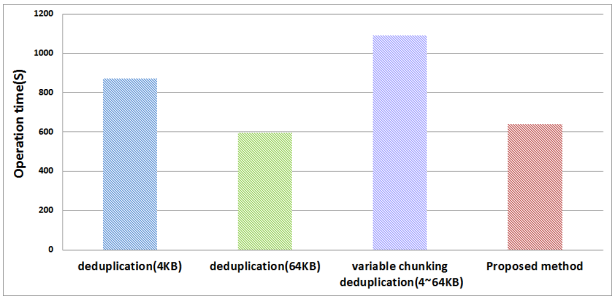

그림 10. 기존의 기법들과 제안한 기법의 연산 시간 비교

Fig. 10. Operation time between existing techniques and the proposed method.

SSD의 장점인 빠른 쓰기 성능을 효율적으로 사용할 수 있다.

전력 소모의 경우 그림 9와 같이 제안한 방법이 4KB에 비해 17% 전력 감소를 보였고, 가변 크기 분할기법에 비해서도 34%의 전력 소모를 감소시키는 성능을 보였다.

그림 10은 제안한 기법과 기존의 기법들과의 연산 시간 비교를 보여준다. 제안한 기법은 64KB의 고정 크기 분할기법과 비슷한 연산 시간을 보였으며, 4KB의 고정 크기 분할기법과 가변 크기 분할기법에 비해 각각 26%와 41%의 성능 향상을 보였다.

V. 결 론

본 논문은 중복 데이터의 응용프로그램 기반 지역성과 유사한 이름을 가지는 지역성 특성의 일치 여부 및 유사비율을 고려하여 청크 크기를 적응적으로 할당하는 지역성 기반 청킹 정책을 제안한다. 중복될 확률이 높은 데이터를 유추하여 청크 크기를 크게 할당함에 따라 중복 제거율은 4KB의 고정 크기분할 기법에 비해 3.2%, 가변 크기분할 기법에 비해 5.9% 감소하였지만, 청킹 및 해시값 생성에 따른 CPU 오버헤드를 줄여 SSD의 쓰기 속도 성능을 4KB의 고정 크기 분할기법과 가변 크기 분할기법에 비해 각각 23.5%, 48%를 향상시켰다. 연산 시간 성능 또한 4KB의 고정 크기 분할기법 및 가변 크기 분할기법과 비교하여 각각 26%와 41%의 성능향상을 보였다. 전력 소모율도 4KB의 고정 크기분할 기법과 가변 크기분할 기법에 비해 각각 17%, 34%를 감소시키는 성능을 확인 할 수 있었다.

참 고 문 헌

[1] J. F. Gantz, C. Chute, A. Manfrediz, S. Minton, D. Reinsel, W. Schlichting, and A. Toncheva, “The diverse and exploding digital universe: An updated forecast of worldwide information growth through 2011,” IDC, An IDC White Paper- sponsored by EMC, March 2008.

[2] D.G. Andersen and S.Swanson, “Rethinking flash in the data center”, IEEE Micro, vol. 30, no. 4, pp.52-54, Jul. 2010.

[3] J. Min et al, “Efficient Deduplication Techiques for Modern Backup Operation,” IEEE TRANSACTIONS ON COMPUTERS, VOL. 60, NO. 6, June, 2011.

[4] Chin-Hsien Wu, Hau-Shan Wu, ”A data de-duplication access framework for solid state drives”, SAC′11, Proceedings of the 2011 ACM Symposium on Applied Computing, pp.600-604, Mar, 2011.

[5] Seung-Kyu Lee, Yu-Seok Yang, Deok-Hwan Kim, “Hybrid Data Deduplication Method for Reducing Wear-Level of SSD-based Server Storage”, Journal of KIISE : Computer Systems and Theory, Vol 38, No 6, pp.292-297, Dec, 2011.

[6] Lawrence You and Christos Karamanolis, “Evaluation of efficient archival storage techniques”, Proceedings of the 21st IEEE / 12th NASA Goddard Conference on Mass Storage Systems and Technologies, pp.1-6, Apr, 2004.

[7] Ahmed El-Shimi, Ran Kalach, Ankit Kumar, Adi Oltean, Jin Li, and Sudipta Sengupta, “Primary Data Deduplication—Large Scale Study and System Design”, Usenix ATC′12, June, 2012.

[8] S. Quinlan and S. Dorward, “Venti: a new approach to archival storage,” in Proceedings of the 1st USENIX conference on File and storage technologies, pp.89-101, 2002.

[9] Athicha Muthitacharoen, Benjie Chen, David Maz Ieres “A low-bandwidth network file system” , in proceeding SOSP ′01 Proceedings of the eighteenth ACM symposium on Operating systems principles. pp.174-187, 2001.

[10] M. O. Rabin, “Fingerprinting by random polynomials”, Center for Research in Computing Technology, Tech. Rep.TR-15-81, 1981.

[11] Yu-Seok Yang, Seung-Kyu Lee, Deok-Hwan Kim, “De-duplication of Parity Disk in SSD-Based RAID System”, Journal of IEEK : CI, acceptance publication, Dec, 2012.

[12] Laura DuBois, Robert Amatruda, “Using Deduplication efficiency & IT cost reduction” IDC analyze the Future. September 2010.

[13] B. Debnath, S. Sengupta, J. Li, “ChunkStash:S peeding up Inline Storage Deduplication using Flash Memory”, USENIX ATC′10, 2010.

[14] A. Gupta, R. Pisolka, B. Urgaonkar, and ASivasubramaniam, “Leveraging value locality in optimizing nand flash-based ssds”, in Proceedings of the 9th USENIX conference on File and storage technologies, 2011.

[15] F. Chen, T. Luo, and X. Zhang, “Caftl: a cont ent-aware flash translation layer enhancing the

lifespan of flash memory based solid state drives" in Proceedings of the 9th USENIX conference on File and stroage technologies, 2011.

저 자 소 개

이 승 규(학생회원)
2011년 대진대학교 통신공학 학사 졸업.
2011년~현재 인하대학교 전자공학과 석사과정
<주관심분야 : 임베디드 시스템, 스토리지 시스템>

김 주 경(학생회원)
2009년 인하대학교 전자공학과 학사 졸업.
2012년~현재 인하대학교 전자공학과 석사과정
<주관심분야 : 임베디드 시스템, 스토리지 시스템>

김 덕 환(정회원)-교신저자
2003년 한국과학기술원 컴퓨터 공학 박사
2006년~현재 인하대학교 전자공학부 교수
<주관심분야 : 시각정보처리, 스토리지 시스템, 임베디드 시스템, BCI>