Journal of the Korea Academia-Industrial cooperation Society
Vol. 14, No. 5 pp. 2449-2456, 2013

http://dx.doi.org/10.5762/KAIS.2013.14.5.2449

대용량 컬럼 저장소를 위한 교차 압축 이중화 기법

변시우[1*]
[1]안양대학교 디지털미디어학과

Cross Compressed Replication Scheme for Large-Volume Column Storages

Siwoo Byun[1*]
[1]**Division of Digital Media, Anyang University**

요 약 컬럼-기반 데이터베이스 저장소는 우수한 입출력 성능으로 대용량 데이터 분석 시스템을 위한 매우 진보적인 모델이다. 전통적인 데이터 저장소는 빠른 쓰기 연산을 위하여 한 레코드의 속성들을 하드 디스크에 연속적으로 배치되어 있는 가로-지향 저장 모델을 활용하였다. 하지만 검색이 대부분인 데이터웨어하우스 시스템을 위해서는 월등한 판독 성능 때문에 컬럼-지향 저장소가 더 적합한 모델이 되고 있다. 또한 최근에는 MLC 플래시 메모리를 사용한 SSD가 고속 데이터 분석 시스템을 위한 적합한 저장 매체로 인식되고 있다.
본 논문에서는 고속 컬럼-지향 데이터 저장소 모델을 도입하고, 고속 컬럼-지향 데이터웨어하우스 시스템을 위한 교차 압축 이중화를 활용하는 새로운 저장소 관리 기법을 제안한다. 본 저장소 관리 기법은 두 개의 MLC SSD에 기반하며, 압축과 비압축된 세그먼트의 교차 이중화를 통하여 높은 CPU 및 입출력 부하에서도 우수한 저장 성능과 안정성을 얻는다. 성능 평가 결과를 통하여 본 저장소 관리 기법이 기존 기법보다 컬럼 세그먼트 갱신 처리치 및 그 응답 시간 측면에서 더 우수함을 확인하였다.

Abstract The column-oriented database storage is a very advanced model for large-volume data analysis systems because of its superior I/O performance. Traditional data storages exploit row-oriented storage where the attributes of a record are placed contiguously in hard disk for fast write operations. However, for search-mostly datawarehouse systems, column-oriented storage has become a more proper model because of its superior read performance. Recently, solid state drive using MLC flash memory is largely recognized as the preferred storage media for high-speed data analysis systems.
In this paper, we introduce fast column-oriented data storage model and then propose a new storage management scheme using a cross compressed replication for the high-speed column-oriented datawarehouse system. Our storage management scheme which is based on two MLC SSD achieves superior performance and reliability by the cross replication of the uncompressed segment and the compressed segment under high workloads of CPU and I/O. Based on the results of the performance evaluation, we conclude that our storage management scheme outperforms the traditional scheme in the respect of update throughput and response time of the column segments.

Key Words : Column-oriented Database, Cross compression, MLC flash memory, SSD Replication

1. 서론

데이터웨어하우스 분야에서 빠르게 부상하고 있는 컬럼-지향(Column-Oriented) 데이터베이스[1,2]는 기존의 한 레코드 단위로 연속 저장하는 가로방향-저장 데이터베이스와는 상반되어, 세로의 필드 단위로 분리, 저장, 검

이 논문은 2013년도 교육과학기술부의 재원으로 한국연구재단의 지원을 받아 수행된 기초연구사업임(No.2013020974)
*Corresponding Author : Siwoo Byun(Anyang Univ.)
Tel: +82-31-467-0922 email: swbyun@anyang.ac.kr
Received March 13, 2013 Revised (1st April 8, 2013, 2nd May 8, 2013) Accepted May 9, 2013

색하는 새로운 데이터베이스이다. 즉, 컬럼-지향 데이터베이스는 세로로 저장하므로, 유사성이 높은 데이터들이 서로 군집되어, 압축, 저장, 검색에 매우 효과적인 구조이다. 이러한 판독 위주의 데이터베이스 환경에서는 특정 컬럼에 데이터가 모여 군집되어 있는 컬럼 저장 구조가 당연히 효율적이며, C-Store[3,4]가 잘 알려져 있다. 또한 최근에는 기업의 중대형 데이터 서버나 데스크톱, 노트북 등 에서 전통적인 하드디스크 드라이브(HDD) 저장 시스템에서 고속 플래시 메모리 기반의 SSD 저장 시스템으로의 대체이동이 빠르게 진행되고 있다[5,6].

기본적으로 세로-저장 스토리지는 테이블 저장시 세로로 분리 저장한다. 예로 들어, 논리적으로 행과 열의 2차원으로 구성되어 있을 경우, 실제 물리적인 저장은 해당 저장 장치에서 연속적인 일차원 바이트 열로 구성되어 운영체제나 버퍼와 연동된다[7].

즉, 세로로 저장하므로, 유사성이 높은 데이터들이 서로 군집되어, 압축, 저장, 검색에 매우 효과적인 구조이다. 이러한 물리적 구조는 저장, 검색 외에도 파티셔닝, 인덱싱, 캐싱에도 성능상 큰 영향을 준다. 과거에는 하드디스크가 대세였고 스토리지도 이에 적응하였으므로, 하드 디스크의 접근 특성을 고려하였고, 당연히 저장 연산 처리에 가장 효율적인 방식인 가로-지향 저장 방식으로 설계되었다. 그러나 이제 고성능 SSD[8,9]가 속도와 더불어 가격 경쟁력을 갖춤에 따라서, 기존의 HDD를 넘어서 차세대 컬럼-지향 저장 시스템으로서 SSD을 활용한 신속한 데이터 검색과 효율적인 스토리지 접근 및 관리 기법이 필요하게 되었다.

2. 제안 연구의 배경

먼저 본 연구의 제안에 앞서, 기초가 되는 새로운 세로-저장 스토리지의 장점을 기존의 가로-지향 스토리지와 비교 분석 하여 보았다.

① 세로-저장 스토리지는 데이터가 메모리에 올라가면, 같은 타입의 데이터가 순차적으로 군집되어 나열되어 있으므로, 당연히 높은 압축 효율과 대용량의 고속 압축이 가능하다. 또한, 이 압축후 접근할 데이터의 크기가 대폭 줄어든 효과로서, 다량의 집합 연산 수행시 일반 스토리지가 불필요하게 전체 레코드를 읽어야 하는데 비하여, 상대적으로 매우 적게(연관된 컬럼만) 읽어도 되므로 매우 I/O 효율적이다[7].

또한, 컬럼-지향 데이터베이스는 실제 SQL 명령 수행시 압축되거나 분리 저장된 데이터가 통합되어 접근이 용이한 레코드 포맷으로 변환 되어야 하는데, 이 변환과정을 실체화, 실자료화(Materialization)[10]라고 한다. 즉, 컬럼별로 분리되어 저장된 데이터를 같은 줄(row)단위로 서로 연결하여, 일반 데이터베이스의 한 투플(레코드)처럼 수평으로 연결하는 작업으로 tuple stitching이라고도 한다. 실자료화를 수행하는 이유는 기존 범용 데이터베이스와 같은 쿼리 인터페이스도 제공하고, ODBC 드라이버와 같은 범용 접속 표준을 제공하기 위함이며, 일반적인 가로-지향 데이터베이스에는 없는 기능이다. 실체화는 변환 시간이 많이 소요되므로, 실체화의 효율과 성능이 좋아야 한다.

② 세로-저장 스토리지는 다량의 레코드에 한 컬럼의 수치를 수정 반영할 때도 상당히 효율적이다. 실제 이런 업데이트 연산이 일상적으로도 자주 발생하는데, 일반 스토리지는 업데이트와 관련 없는 다른 컬럼들을 어쩔 수 없이 스토리지로부터 읽고 메모리상에도 할당시키느라 엄청난 시간과 공간을 낭비한다. 반면에 세로-저장 스토리지는 연산과 무관한 컬럼은 아예 터치하지 않고, 관련된 컬럼에만 압축된 최소한의 수정연산만 수행하므로, 상당히 공간 효율적이며, 연산 속도도 매우 빠르다.

③ 반면, 일반 스토리지의 장점은 한 레코드의 많은 컬럼이 수정되는 경우나 새로운 데이터가 빈번히 추가되는 경우, 즉 온라인 쓰기 연산이 빈번한 경우에 효과적이다. 즉, 읽기가 적고, 쓰기가 매우 많은 경우는 세로-저장 스토리지가 불리한 조건이 된다.

현재 하드 디스크는 저장 용량이 큰 반면, 속도가 느리며, 전력 소모가 평상시에도 전체 시스템 전력의 20%정도로 매우 큰 편이며, 사용 대기 중에도 전력이 계속 소모된다. 이에 대한 대안으로 플래시 메모리를 장착한 SSD가 있으며, 이미 고속화, 대용량화 추세에 있다. 다만, 일반 메인 메모리와는 달리, 쓰기와 소거 연산에 상당히 많은 시간이 더 소모되며, 쓰기 횟수가 제한되는 고유한 특성이 있다. 이 수명을 넘기면 더 이상 쓰기 연산을 수행할 수 없다. 단, 판독 연산은 계속하여 수행이 가능하다[11]. 또한, 플래시는 일반 메인 메모리와 달리 바로 쓰기 연산을 수행할 수 없고, 이전에 미리 블록 단위로 포맷 개념의 소거 연산을 수행한 후 쓰기 연산을 수행할 수 있다. 즉, 플래시 메모리는 PC의 메인 메모리처럼 Update-In-Place(제자리 덮어 쓰기)가 불가능한 Update-Out-Place 구조이다[5].

세로-저장 스토리지도 시스템 운영을 위하여 반드시 저장 장치가 필요하다. 물론, 이러한 저장 장치로서 기존의 전통적인 하드디스크를 그대로 사용할 수도 있다. 하지만, 본 연구에서 새로운 저장기술인 MLC 플래시 메모리 저장 장치를 최신 컬럼-지향 데이터베이스에 접목시키면, 다음과 같은 이유로 최상의 효율과 성능을 만들 수

있음을 주장한다.

먼저, 플래시 메모리 SSD는 그동안 고가였던 이유로 검토 대상에서 제외되었지만, 현재는 MLC SSD는 충분한 가격경쟁력을 갖추고 있음이 첫째 이유이다. 둘째, 플래시 메모리가 최근 빠른 추세로 대용량화 되고 있으며, 판독 속도가 하드디스크에 비하여 매우 빠른 장점이 있다. (보통 250MB/s로서 3배 이상 빠름) 따라서, 세로-저장 스토리지의 사용처가 대부분 고속 읽기가 필요하므로 매우 적합하다.

이러한 관점에서, 본 연구에서는 컬럼-지향 데이터베이스의 운영 특성을 효과적으로 활용하며, 고속 MLC SSD의 장점을 최대한 반영하고자 한다. 또한, CPU 및 I/O의 높은 부하와 변동성에서도 대용량 초고속의 컬럼-지향 데이터베이스에 최적화된 저장 성능과 안정성을 확보할 수 있는 효율적인 교차 저장 기술을 제안하고자 한다.

3. 본 론

3.1 컬럼–지향 데이터베이스를 위한 효율적인 저장 시스템

컬럼-지향 데이터베이스는 고속으로 대용량의 데이터를 처리하기 위하여 하부 저장시스템에 압축하여 저장한 후, 필요시 압축을 복원하는 실체화 과정을 거쳐서 메모리상에서 고속으로 입출력 연산이 수행된다. 이를 효과적으로 안정적으로 지원하기 위해서는 고속 저장이 가능하면서도 사후 복구가 아닌 저장 매체의 장애를 미리 예방할 수 있어야 한다.

특히, 컬럼-지향 데이터베이스는 매우 큰 대용량 데이터베이스를 사용하며, 그 내용도 사업전략 및 마케팅 실적과 같은 의미 분석용이므로 데이터가 복잡하며, 의사결정 등의 시급한 업무를 수행한다. 따라서 백업 데이터를 로딩이후 다시 의미를 분석하고 의사결정을 계속 하려면 시간적 오버헤드가 너무 크므로, 데이터를 이중화 시켜서 장애 및 고장을 사전에 예방하여야 한다. 즉, 대용량 컬럼-지향 데이터베이스의 저장 매체의 복구에는 복잡하고 시간이 많이 소모되므로, 저장 시스템에서 한 카피 이상을 중복 저장하여 데이터베이스의 신뢰성을 높이는 방법이 필요하다.

이러한 신뢰성을 높이는 보편적 방법으로 두 개의 카피가 항상 동일한 단순하며 물리적인 미러링 기법이 사용되어 왔다. 미러링은 각 동기 장치의 쓰기가 완전히 둘 다 완료되어야 저장 연산이 종료된다. 본 제안 기법에서는 이중화란 용어를 사용하는데, 논리적으로 미러링 개념과 비슷하기는 하지만, 이중화는 두 카피가 완벽히 물리적으로 동일할 필요는 없으므로, 본 연구에서는 이를 사용하기로 한다.

3.2 교차 압축 이중화 기법의 제안

본 제안 기법에서 이중화의 원리는 두 카피가 동일한 데이터를 유지하지만, 하나의 컬럼-세그먼트 단위로 Fig. 1과 같이 번갈아 교차하여, 압축이 되지 않은 원래의 데이터와 압축이 수행된 데이터를 교차하여 저장한다. 또한, 하나의 카피만 완성되면 다른 하나의 카피가 완성될 때 까지 기다리지 않고, 비동기 개념으로 저장 연산을 리턴 한다. 물론, 물리적인 미러링 기법에 비하면, 이론적인 안정성은 더 낮다. 하지만, 컬럼-지향 데이터베이스는 램 메모리에 실체화를 거친 데이터가 지속적으로 존재하며, 이를 이중화 장치로 보내는 것이므로, 램과 두 MLC에 상시 데이터가 존재하게 되므로 현실적인 안정성은 유지된다.

이중 교차 압축에서 사용되는 압축 방식은 소스가 GPL로 공개되어 확보가 용이하고, 압축효율이 높은 LZO 기법[12]를 사용하였다. 또한, 미러링이 압축된 카피 두 개를 물리적으로 운영함과는 달리, 본 방법은 압축과 비압축의 두 버전을 운영한다. 다른 운영 작업등의 이유로 CPU가 busy인 상태가 되면, 복원속도가 떨어져서 실체화가 느려지는 상황이 발생하는데, 이 경우에는 비압축 버전을 판독함으로써 실체화 성능을 높일 수 있다.

또한, 하드디스크로 풀 백업을 해야 할 경우에는 압축과 비압축 버전중 하나를 융통성 있게 선택하여 디스크로 백업할 수 있다. 이 경우 압축백업이 속도측면에서는 더 빠르겠지만, 컬럼-지향 데이터베이스와는 다른 상업용의 일반 데이터베이스와 연동이나 상호 데이터 변환을 위해서는 비압축의 일반 포맷의 버전으로 저장할 경우 비압축버전이 필요하다.

요약하면, 본 기법은 컬럼-지향 스토리지로서의 저장 성능에서도 우세하며, 디스크 백업의 유연한 선택성도 부여할 수 있으며, CPU 자원이 부족시 비압축의 빠른 실체화도 가능하다. 물론 CPU 자원이 충분하면, 교차 식으로 압축데이터를 판독하면 되므로, 압축 미러링 방식의 판독 방법과 같아지므로 그 속도도 같다.

다만, 교차 압축 방식에서는 비압축 데이터도 저장해야 하므로, 압축 미러링에 비하여 더 많은 저장 영역을 차지하지만, MLC 스토리지의 지속적인 대용량화와 고속화 및 가격하락으로 가격대비 성능이 전체적으로 유리하다.

컬럼-지향 데이터베이스의 성능 개선을 위하여 고려하

여야 할 또 다른 중요한 논점은 CPU 및 저장 부하의 변동성이다. 특히, 일반 온라인 데이터베이스에 비하여 컬럼-지향 데이터베이스의 데이터웨어하우스는 소수의 인원이 집중적인 피크 로드를 발생했다가, 유휴 시는 CPU 로드가 매우 급격히 감소하므로, 이 변동성이 훨씬 더 크다.

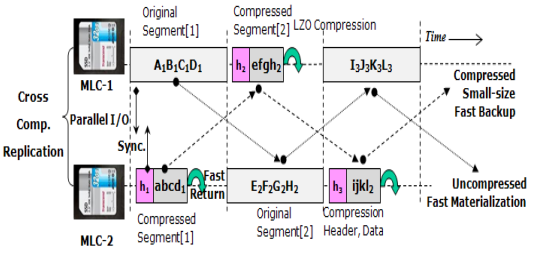
따라서 일반적으로 단점으로 보는 이 변동성을 역이용하여, CPU 사용 특성을 잘 고려하면 매우 효과적인 저장 시스템을 설계할 수 있다. 즉, CPU 자원이 충분할 때는 저장장치의 I/O가 느리므로, 압축하여 컬럼-세그먼트를 저장함이 매우 효율적이다.

하지만, 컬럼-지향 데이터베이스의 특성상 CPU 기반의 대용량 분석 작업과 RAM과의 CPU간의 데이터 처리 등으로 고부하 집중작업시 CPU는 압축연산에 충분한 CPU 자원 제공이 어려워진다. 이 경우 압축하지 않고 그냥 비압축 세그먼트를 저장하는 것이 더 빠르고 효율적이며, 압축에 대한 CPU 자원을 기존 분석 작업에서 빼앗지 않아서 사용자의 데이터 분석 작업 속도도 저하되지 않다.

그리고 MLC-SSD 측면에서도 사용자의 컬럼 읽기나 저장 연산이 집중되는 경우, 평소속도보다 수 배 이상의 느린 적체 현상이 발생할 수 있다. 특히 큰 컬럼 데이터를 저장 하는 도중에 심한 경우 수 초 씩 정지될 수 있다. 이러한 심한 I/O 적체 상황에서는 CPU 자원을 최대한 활용하여 압축 저장하는 것이 더 효율적이다.

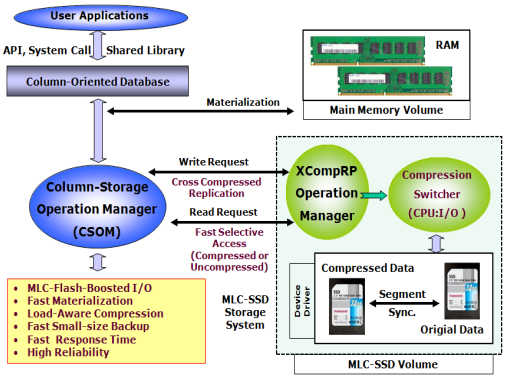
또한, 컬럼 세그먼트를 교차 압축시키면 부수적인 효과로서, 빠른 백업과 고속 실체화의 효과를 얻을 수 있다. 즉, Fig. 1에서처럼 압축된 세그먼트의 점선 링크를 따라가면서 백업을 진행하면 압축된 일련의 백업 스트림이 구성되므로 압축시간을 다시 소모하지 않고 고속 백업이 가능하다.

마찬가지로 비압축 세그먼트의 링크를 따라가면서 컬럼 데이터베이스의 레코드들을 실체화시키면 압축복원과정을 거치지 않고 바로 램 메모리로 복사가 가능하므로 빠른 실체화가 가능하다.



[Fig. 1] Architecture of Cross Compressed Replication

결론적으로 본 제안 기법을 통하여 대용량 고속 분석이 필요한 컬럼-지향 데이터베이스 환경에서 이러한 부하 변동성을 효과적으로 고려하며, 압축 및 비압축 세그먼트의 교차 압축을 통한 저장 시스템의 성능과 안정성을 모두 높일 수 있다. 아래 그림은 본 제안 기법의 시스템 구조도이다.



[Fig. 2] Architecture of Proposed System

먼저, 사용자나 응용프로그램의 입출력 요청이 컬럼-데이터베이스 시스템에 접수되면, 그림 중간의 컬럼 저장 연산 관리자(Column-Storage Operation Manager: CSOM)를 거쳐서 하부의 교차 압축 이중화 관리자(XCompRP Operation Manager)로 전달된다. 교차 압축 이중화 관리자는 압축과 비압축 모드로 컬럼-세그먼트를 연속적으로 교차 압축하는 저장 기능을 주관하며, CPU 및 I/O 부하를 고려하여 신속하게 압축 과정이 물리적으로 하부 MLC-SSD에 반영되게 한 후, 처리된 결과를 리턴 한다.

컬럼-지향 데이터베이스의 저장 성능을 좀 더 높이기 위하여, 추가적으로 컬럼의 압축 특성을 활용할 수 있다. 폴라라이징이라고 하는 스토리지 관리 기법인데, 이는 한 테이블 컬럼을 활성-컬럼과 비활성-컬럼으로 양분하여 압축 저장한다, 즉, 저장 요청이 발생된 컬럼 데이터 중에서, 컬럼의 압축률이 높으며 빈번히 쓰기가 이루어지는 컬럼인 고활성-컬럼과 압축이 잘 안되고 접근이 되지 않는 저활성-컬럼으로 양극화시켜서 분류하여 저장하여 관리한다. 반면, 본 제안기법에서는 양극화 없이 압축과 비압축 버전을 이중으로 저장한다. SSD를 구성하는 플래시 메모리는 업데이트가 빈번한 고활성-속성의 컬럼-데이터들을 같은 저장 영역으로 군집화를 유도하면, 개별적으로 수행해야 할 쓰기와 소거 연산을 집합 개념으로 한 번에 몰아서 수행 가능하므로, 전체적인 저장-연산의 오버헤드를 줄일 수 있다. 특히, 컬럼-데이터의 군집화는 일반적으로 마구 섞여 있는 랜덤 저장의 경우보다 상대적으로 소

거오버헤드가 줄어서 결국은 SSD의 수명도 연장된다[2].

3.3 제안 기법의 수행 알고리즘

제안 기법인 XCompRP(Cross Compressed RePlication)의 수행 알고리즘은 다음과 같다.

① 사용자의 프로그램이 이중 교차 압축된 컬럼-저장 스토리지에 대한 판독 또는 쓰기 연산(operation)을 요청한다.

② 사용자의 입출력 요청이 접수되면, 요청된 해당 연산은 XCompRP의 이중화된 읽기 또는 쓰기 연산으로 전환된다. 이를 관리하는 Column-Storage Operation Manager(CSOM)는 입출력 요청 연산을 자신의 컬럼-저장 연산-큐의 마지막에 삽입한다. 추후에 방금 요청한 컬럼-저장 연산이 완료되면 그 처리 결과 값을 사용자 프로그램에 최종적으로 전달하며 종료된다.

③ 컬럼-저장 연산-큐에서 한 연산씩 차례로 꺼낸 후 그 연산의 종류를 분류한다. 컬럼-읽기 연산이면, ④XCompRP_READ을 수행하며, 쓰기 연산이면 ⑤XCompRP_WRITE를 수행한다.

④ XCompRP_READ:

1) 컬럼-지향 데이터베이스 시스템의 가용할 CPU 자원이 충분할 경우에는, 압축된 컬럼-데이터가 판독 속도가 더 빠르므로 압축-세그먼트를 저장한 MLC-SSD에 판독 연산을 포워딩한다. 또한, 저장 장치 의 병목현상으로 판독 연산들이 적체되는 경우에도 동일하게 포워딩한다. 컬럼-지향 데이터베이스는 고압축 특성을 가지며, MLC-SSD는 압축된 블록을 속도가 매우 빠르다.
2) 만일, 가용할 CPU 자원이 부족하여 압축데이터의 읽기가 심하게 적체되거나 1)에서 판독 오류가 발생하면, 비압축된 컬럼-세그먼트를 저장한 다른 MLC-SSD로 부터 컬럼-데이터를 판독한다.
3) 위의 MLC-SSD로부터 판독 작업이 성공하면, 판독한 값을 CSOM에게 전달한다.

⑤ XCompRP_WRITE:

1) 이중 압축 교차 쓰기에 해당하는 저장 연산으로서 다음 두 저장 연산이 병렬로 동시에 실행한다. 즉, 한 MLC-SSD의 컬럼-세그먼트는 비압축으로 원래의 컬럼-데이터를 저장하며, 다른 MLC-SSD는 LZO 압축 방식을 기반으로 컬럼-세그먼트를 압축하여 저장한다.
2) 만일 컬럼 데이터의 저장분량이 한 세그먼트의 길이를 초과하면 두 MLC-SSD가 세그먼트 저장 모드를 바꾸어 같은 방식으로 압축 또는 비압축으로 번갈아 가며 계속 저장한다.
3) 두 MLC-SSD 중에 한 저장소라도 컬럼-저장 연산의 수행이 정상적으로 완료되었다는 이벤트가 들어오면, 처리 시간을 기록하고, CSOM에게 컬럼-저장 완료 메시지를 리턴 한다. 다른 MLC-SSD는 계속 저장하여 완료한다.

4. 성능 평가

제안기법과 비교된 저장 관리 기법은 MLC 저장 장치를 1개만 사용하는 단일 비압축 저장 기법(UnComp_1), 단일 압축 저장 기법(Comp_1)과, 가장 보편적인 복제 구성인 압축 미러링 기법(Comp_MR)이다.

4.1 실험 환경

1) 실험 시스템 구축 환경 및 평가 지표

하드웨어 환경은 인텔 i7 CPU에 메인 메모리 8G, 운영체제는 64bit 윈도우즈7을 사용하였다. 또한, 총 1억 건의 데이터베이스를 구축하였으며, 성능 수치 분석용으로 CSimAPI[13]가 사용되었다. 이 모듈들을 통하여 원하는 만큼이 일정한 부하가 생성되므로, 초당 일정한 수의 컬럼 세그먼트 저장 연산을 발생시킨 후, 이 요청을 수행한 저장 시스템의 처리 시간과 성능을 측정하면 된다.

주요 성능 관련 평가 지표는 컬럼 세그먼트 저장 연산 처리치(column-segment update throughput)와 그 응답 시간(response time)이다. update throughput은 초당 몇 개의 저장 오퍼레이션이 처리되었는지를 의미하고, response time은 컬럼 저장 오퍼레이션이 발생한 후 수행완료까지 걸린 시간을 의미한다.

2) 실험용 컬럼 데이터베이스의 구성 및 압축 특성

본 실험에 앞서서, 국내 한 기업의 테이블과 운영 데이터를 활용하여 총 1억 건의 데이터를 구축하였다. 각 컬럼을 압축하기 위하여, LZO 압축 저장 기법을 사용하였는데, 컬럼-저장의 군집 특성상 압축률이 매우 좋은 편이다.

특히, 종류가 한정된 컬럼 및 불리언 타입의 컬럼이나, NULL 값이 많은 경우 압축이 가장 잘되어 2%로서 거의 다 압축되기도 하였다. 그 다음으로 일련번호, 사용자 번호가 5%대로 압축이 잘되며, 날짜와 초단위로 표시되는 시간속성의 경우에는 10%정도로 압축이 되었다. 주소 속성과 같은 불규칙한 문자열이 많이 포함되는 경우는 27%로 압축되었다.

텍스트 메모성의 랜덤한 긴 스트링 데이터가 있는 경우는 압축효율은 더욱더 낮았다. 실제 테스트 해보니 약 53%로 압축되었다. 압축 속도는 본 실험환경을 기준으로 느린 경우 초당 약 3Mbyte에서 최대 약 30Mbyte로 컬럼 특성에 따라 매우 가변적이었다.

컬럼 데이터를 압축 저장시 대용량의 컬럼 데이터를 통째로 저장하면, 압축 시간도 매우 느리고, 복원 시간도 매우 느리게 되므로, 적당한 양의 컬럼 세그먼트로 단위로 나누어서 저장하게 하였다.

본 실험에서는 1000건 단위로 컬럼 세그먼트를 구성하였으며, 데이터베이스의 일반적인 데이터의 평균 크기가 20바이트 이하이므로, 평균적인 20% 압축비를 고려하면, 한 세그먼트는 약 4KB 크기로 저장 된다. 본 실험에 사용되는 MLC SSD의 파일 시스템의 최소 파일 점유 크기인 4KB인 점을 고려하여 맞추었으며, 이 크기의 배수가 되므로 적절하다.

또한, 압축률과 압축속도가 컬럼 특성에 따라 매우 가변적임을 고려하여, 실험의 공정성을 위하여 최대 최소 범위에서 임의로 선택되게 하였다. 그리고 사용 가능한 CPU 자원에 따라 압축 속도가 크게 달라지므로 10%~100% 범위에서 CPU 자원이 변화되도록 변동성을 충분히 고려하였다.

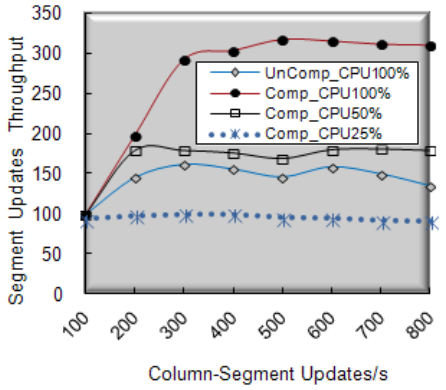
4.2 압축 특성 및 부하 변동성 분석 실험

먼저, CPU 자원의 가동률에 따른 성능 변화를 파악하고자 본격 실험에 앞서서 아래와 같이 예비 테스트를 수행하여 보았다.

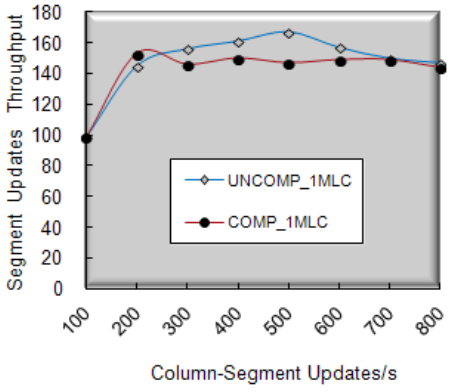
그래프에서 UnComp_CPU100%는 비압축방식으로 CPU가 100% 지원될 때의 컬럼 세그먼트 저장 성능 곡선이며, Comp_CPU100%, 50%, 25%는 압축방식으로 CPU가 각각 100%, 50%, 25%만 지원될 때의 처리 성능을 보여준다. Fig.3과 Fig.4의 가로축은 컬럼-세그먼트 저장 시스템에 가해지는 저장 연산의 부하를 의미하며, 세로축은 그 부하에서의 처리된 컬럼-세그먼트 저장 연산수를 의미한다.

먼저, Fig. 3을 보면, CPU 가동률이 약 40%이하 구간에서는 컬럼 세그먼트 저장 방법이 단순 비압축 저장 방식 보다 압축 방식이 더 저장 성능이 낮다. 대략 비압축 방식이 38%이상 성능이 더 높으며, 압축 방식에 비하여 CPU도 훨씬 덜 사용되므로 시스템에 더 효율적이다. 반대로 40% 이상의 CPU 자원이 확보되면 압축 방식이 더 효율적으로 바뀐다.

본 예비 실험을 통하여, 결론적으로 CPU 자원의 변동성이 시스템 저장 성능에 매우 큰 영향을 미침을 알 수 있었다. 즉, CPU가 충분하지 않을 경우 비압축 방식이 더 효과적일 수 있고, 충분할 경우는 압축방식이 더 효과적일 수 있다. 이러한 변동성은 의미 분석용으로 CPU를 집중적으로 많이 쓰는 컬럼-지향 데이터베이스 환경에서 중요한 변수이다.



[Fig. 3] Throughput by CPU allocation



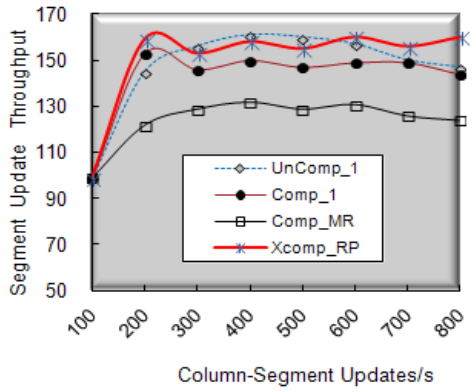
[Fig. 4] Throughput by CPU variation

Fig. 4는 이러한 CPU 자원 변동률을 10~100% 까지 랜덤하게 변형시키면서, 압축 저장과 비압축 저장의 효율을 비교한 것이다. 먼저, 비압축방식인 UnComp_1MLC은 CPU 자원을 거의 안 쓰며, 중저부하 구간에서 잘 견디다가, 결국 고부하 구간에서 MLC I/O의 저장 부하 증가로 인한 쓰기 적체로 성능저하가 발생하였다. 압축방식인 Comp_1MLC도 저부하 구간에서는 적은 CPU 자원으로도 잘 견디다가, 중부하 구간부터 CPU 자원 부족으로 처리 성능이 올라가지 못하고 역시 하강하며 정체되었다.

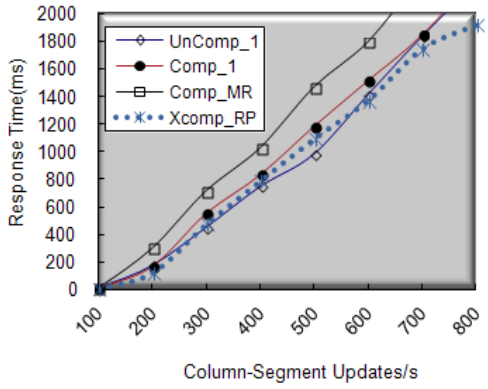
결론적으로 두 방법 모두, 고부하 구간에서는 MLC I/O 부하나 CPU 부하의 둘 중 하나의 이유로 모두 저장 성능이 낮아졌다. 그런데, 만일 두 개의 방식을 잘 혼용하여 동시에 가동하고, 그 두 원인중 하나를 최대한 회피하

고 더 우세한 방식을 택하면, 전체적으로 더 좋은 성능을 고부하 구간에서도 발휘할 수 있다.

4.3 제안 기법의 성능 실험 및 비교



[Fig. 5] Throughput of Column Updates



[Fig. 6] Response Time of Column Updates

이 실험은 초당 발생된 컬럼 세그먼트 저장 연산의 수가 스토리지 시스템 성능에 어떤 영향을 미치는 지를 분석하기 위함이다. Fig. 5에서 보면, 전반적인 컬럼-세그먼트 저장 연산의 처리 성능을 측정한 결과, 제안기법인 Xcomp_RP가 타기법인 UnComp_1, Comp_1, Comp_MR 보다 높게 나타났다.

저부하 구간에서는 CPU 부하와 MLC I/O 부하가 시스템 안에서 충분히 수용되어 성능 저하를 일으키지 않는다. 하지만, 생성되는 컬럼-세그먼트 저장 연산의 수가 점차로 증가하면, 중간 부하 구간부터 저장 연산들이 작업 큐에 과도하게 쌓이면서, 컬럼-세그먼트 저장 연산 처리가 지체되며 성능이 점차 저하되게 된다.

하지만, 동일한 워크로드 조건에서도 Xcomp_RP가 타기법에 비하여 성능이 상대적으로 더 높았다. 평균하여, Xcomp_RP가 기존 Comp_MR 기법보다 최소 23.4% 이상의 높은 처리 성능을 보였다. 특히, 최고부하 구간에서의 저장 연산 처리치는 Xcomp_RP가 기존의 Comp_MR 보다 29.0%정도 더 높았다. 그 이유는 위의 저장 성능 저하의 주요 원인인 CPU의 부하 및 MLC의 쓰기 연산의 부담에 의한 저장 적체 현상을 교차 압축 이중화 기법에서는 한 MLC에 우선 저 장후 신속히 리턴 함으로써 적체시간을 단축시킬 수 있었기 때문이었다.

즉, CPU와 MLC I/O의 변동성에 능동적으로 반응하여 Comp_MR보다 신속하게 완료하였기 때문이다. 반면, Comp_MR는 압축된 두 개의 카피가 MLC에 모두 저장되어야만 리턴 되므로 둘 중 하나만 적체되어도 전체 응답시간은 단축되지 않으며, CPU가 적체되는 상황에서 MLC I/O까지 적체될 경우 상당한 지연이 발생한다.

Fig. 6을 통하여 응답시간을 비교해보면, 응답속도의 순서가 Fig. 5의 성능순서와 일치하였다. 저부하 구간에서는 단일 MLC SSD로 컬럼-세그먼트를 압축하여 사용하는 Comp_1 기법이 비압축된 UnComp_1 보다 처리 성능과 응답시간이 더 높게 나타났으나, 중부하 구간이후에는 오히려 Comp_1의 성능이 더 낮아진다.

그 이유는 Comp_1은 컬럼-세그먼트의 저장 부하가 많이 걸릴수록 CPU의 압축 부하도 더 심하게 걸리면서, 이에 따른 적체 현상도 발생하는 반면, UnComp_1은 CPU의 압축부하가 없는 비압축저장이므로 상대적으로 더 유리하기 때문이다. Fig. 6의 응답시간 측면을 분석하면, 전체 구간에서 Xcomp_RP가 기존 Comp_MR을 포함한 기존의 기법들보다 최소 51.4% 이상 응답시간이 더 개선되었음을 알 수 있었다.

5. 결 론

세로 저장 방식의 컬럼-지향 데이터베이스는 대용량 데이터의 고속 분석용으로 적합한 새로운 저장 모델이다. 본 연구에서는 컬럼-지향 데이터 모델에 플래시 메모리 기반의 고속 MLC 저장 기술을 융합하였으며, 고속 컬럼-기반 저장소의 성능 개선과 안정성 향상을 위하여, 교차 압축 이중화를 활용하는 새로운 저장 관리 기법인 XcompRP를 제안하였다.

XcompRP 기법은 두 개의 MLC SSD에 기반하며, 압축과 비압축된 세그먼트의 교차압축을 통하여 높은 CPU 및 MLC 입출력 부하에서도 우수한 저장 성능과 안정성을 얻을 수 있다. 또한, 일억 건의 데이터를 기반으로 저장 워크로드를 부가하여, 본 제안 기법이 기존 MLC 미러

링 기법보다 더 우수함을 확인하였다.

실험 분석 결과를 통하여, 전체 부하 구간에서 평균하여, XcompRP가 기존 기법보다 최소 23.4% 이상의 높은 저장 성능을 보였으며, 그 응답시간 측면에서도 51.4% 이상 개선되었다.

향후 SSD 스토리지와 컬럼-지향 데이터베이스의 안정성 및 속도 개선을 위하여 더 효율적인 압축 알고리즘을 적용하고, SLC 및 다양한 스토리지를 활용하며, 특정 컬럼에 대한 선별적인 지연 이중화 및 빠른 복구 방법론을 연구할 계획이다.

References

[1] D. Abadi, S. Madden, and M. Ferreira. “Integrating compression and execution in column-oriented database systems”, *Proc. of SIGMOD,* pp. 671 – 682, 2006.
DOI: http://dx.doi.org/10.1145/1142473.1142548

[2] S. Byun. “Column-aware Polarization Scheme for High-Speed Database Systems”, *Journal of Korean Socieity Internet Information,* Vol. 13, No.3, pp. 83 – 91, 2012.
DOI: http://dx.doi.org/10.7472/jksii.2012.13.3.83

[3] D. Abadi, A. Boncz, and S. Harizopoulos, “Column-oriented Database Systems”, *Proc. of the VLDB*, Lyon, France, August 24-28 2009.

[4] S. Harizopoulos, V. Liang, D. J. Abadi, and S. Madden, “Performance tradeoffs in read-optimized databases”, *Proc. of* VLDB, pp. 487 – 498, 2006.

[5] S. Byun. “Search Performance Improvement of Column-oriented Flash Storages using Segmented Compression Index”, *Journal of the Korea Academia-Industrial,* Vol. 14, No.1, pp. 393 – 401, 2013.
DOI: http://dx.doi.org/10.5762/KAIS.2013.14.1.393

[6] Solid Data Systems, "Comparison of Drives Technologies for High-Transaction Databases", Solid Data Systems, Inc. White paper, 2007

[7] A. Halverson, J. Beckmann, and J. Naughton. "A comparison of c-store and row-store in a common framework", Technical Report, UW Madison Department of CS, TR1566, 2006.

[8] Lucas Mearian, “Analysis: SSD performance -- is a slowdown inevitable?”, Available From: http://www.computerworld.com/s/article/9132668/Analysis_SSD_performance_is_a_slowdown_inevitable_?taxonomyId=19&pageNumber=3, (accessed 16 Mar. 2013)

[9] Samsung, Samsung, what is NAND Flash based SSD?,Available From: http://www.samsung.com/global/business/semiconductorproducts/flash/Products_FlashSSD.html, (accessed 16 Mar. 2013)

[10] D. Abadi, D. Myers, D. DeWitt, and S. Madden. “Materialization strategies in a column-oriented dbms”, MIT CSAIL Technical Report. MIT-CSAIL-TR-2006-078, 2006
DOI: http://dx.doi.org/10.1109/ICDE.2007.367892

[11] S. Byun, M. Hur, and H. Hwang, "An index rewriting scheme using compression for flash memory database systems" *Journal of Information Science*, Vol. 33, No.4, pp. 398 – 415, 2007.
DOI: http://dx.doi.org/10.1177/0165551506076331

[12] Oberhumer, LZO-- a real-time data compression library, Available From: http://www.oberhumer.com/opensource/lzo/lzodoc.php, (accessed 16 Mar. 2013)

[13] Mesquite, CSIM2.0 Development Toolkit for Simulation and Modeling, Available From: http://www.Mesquite.com/documentation/documents/CSIM20_User_Guide-C.pdf, (accessed 16 Mar. 2013)

변 시 우(Siwoo Byun) **[정회원]**

- 1989년 2월 : 연세대학교 전산과학과 (공학사)
- 1991년 2월 : 한국과학기술원 전산학과 (공학석사)
- 1999년 8월 : 한국과학기술원 정보및통신학과 (공학박사)
- 2000년 3월 ~ 현재 : 안양대학교 디지털미디어학과 교수

<관심분야>
데이터베이스, 저장장치, 스마트 임베디드 시스템