제33회 한국정보처리학회 춘계학술발표대회 논문집 제17권 제1호 (2010. 4)

동적 모드 변경을 제공하는 중복 제거 서버

정호민*, 김진, 고영웅
한림대학교 컴퓨터공학과
e-mail:{chorogyi, jinkim, yuko}@hallym.ac.kr

Deduplication Server Supporting Dynamic Mode Change

Ho Min Jung*, Jin Kim, Young Woong Ko
Dept of Computer Engineering, Hallym University

요 약

현재 중복 제거 기술은 클라이언트 기반 중복 제거 모델, 인라인(in-line) 중복 제거 모델 그리고 포스트 프로세스(post-process) 중복 제거 모델로 구분할 수 있다. 본 연구에서는 클라이언트와 서버의 부하를 모니터링하여 시스템 부하에 따라 중복 제거의 핵심 작업을 동적으로 변경한다. 즉, 클라이언트가 유휴하고 서버의 자원 사용량이 높은 경우에는 클라이언트 기반 중복 제거 모델로 동작시키고, 클라이언트의 자원 사용량이 높고 서버가 유휴한(idle) 경우에는 인라인 중복 제거 모델로 동작시킨다. 그리고 전체 시스템이 과부하인 경우는 포스트 프로세스 모델로 동작하게 된다. 제안하는 방식에 대한 실험 결과 전체 시스템의 처리율이 높아지는 것을 확인하였다.

1. 서론

최근 컴퓨터를 이용한 문서 작업이 증가하고, 인터넷을 이용하여 데이터를 교환하는 것이 일반화되면서 저장 시스템(storage system)에 대한 요구 사항이 지속적으로 중요해지고 있다. 대부분의 저장 시스템은 사용자의 요구에 의해서 다양한 종류의 파일을 일정한 크기의 블록으로 변환하여 저장을 하게 된다. 이에 따라 회사에서는 좀 더 많은 저장 공간이 필요하게 되며 이를 유지보수 하는 비용이 늘어나게 되었다. 특히 파일을 저장하고 서비스 해주는 P2P(peer-to-peer)사이트, 엄청난 용량들의 3차원 모델링 자료를 처리하는 개발 서버들은 백업과 안전한 저장을 위해 저장 공간을 많이 필요로 하게 된다. 그래서 이를 극복할 기술들 (압축, 싱글 인스턴스 스토리지, 씬 프로비저닝) 은 이미 널리 쓰이고 있으며 그 중에서도 중복 제거 기술은 놀라운 효율성을 제공한다.

중복 제거 기술은 해시 함수를 사용해 파일들의 각 부분, 블록의 고유한 아이디를 얻어내 이를 비교하여 중복 여부를 판단하는 것이다. 여러 스토리지 시스템에서 높은 비율의 중복 데이터가 발생하고 있으며 이는 원본과 여러 복사본이 존재한다는 의미이므로 고유한 아이디비교를 통해 원본만 저장하는 것이다.

이러한 중복 제거 기술은 클라이언트 기반 중복 제거 모델, 인라인(in-line)[1][2] 중복 제거 모델 그리고 포스트 프로세스(post-process)[3] 중복 제거 모델로 구분할 수 있다. 인라인 모델은 서버에 부하가 몰릴 경우 실시간으로 처리할 수 없으며 포스트 프로세스 모델은 신속하게 데이터를 서비스 할 수 없어 실시간적인 처리에 적합하지 않다.

본 연구에서는 클라이언트와 서버의 부하를 모니터링하고 중복 제거의 핵심 작업을 시스템 부하에 따라 동적으로 변경하는 모델을 제시한다. 서버의 부하가 높을 경우 포스트 프로세스, 클라이언트 기반 모델로 동작하게 되고 서버의 부하가 작을 경우에는 인라인 중복 제거 모델로 동작 시키는 것이다. 또한 서버에서 처리하는 작업 중에 부하가 큰 해싱 작업을 클라이언트에서 전담하여 중복제거 서버의 부하를 줄이는 것을 제안한다.

본 논문의 순서는 다음과 같다. 2장에서는 관련 연구 동향에 대해서 기술하고, 3장에서는 시스템의 설계 및 구현 세부 사항에 대해서 기술한다. 마지막으로 4장에서 결론을 맺는다.

1) 본 연구는 중소기업청에서 지원하는 2009년도 산학연공동기술개발사업 (No. 00039553)의 연구수행으로 인한 결과물임을 밝힙니다.

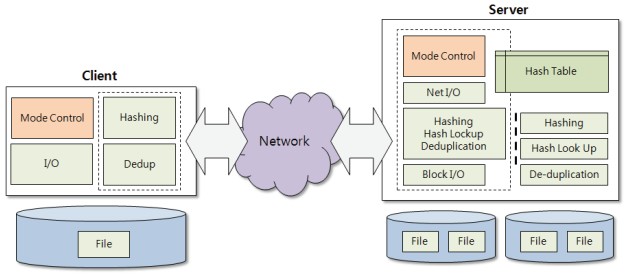
2. 관련 연구

중복 제거의 특징으로는 디스크에 저장되는 용량을 줄이고 네트워크 밴드위스를 절약하며 심지어 문서의 중복된 부분을 찾는데 이용된다. 중복 제거 기술은 네트워크 자원을 효율적으로 사용하더라도 반드시 데이터의 동기화를 유지해야 한다.

중복 제거 기술의 대표적인 연구로는 밴티(Venti)[4]와 저 대역폭 네트웍 파일 시스템(Low-Bandwidth Network File System - 이하 LBFS)[5]이 있다. Venti는 파일을 고정된 크기(8Kbyte)의 블록으로 나누고 각 블록에 SHA1[6] 해시를 적용하여 해시의 값을 비교하여 중복을 판단한다. LBFS는 파일 전송 효율을 높이기 위해 CDC(Content-defined Chunks) 방식을 사용하는 방식이다. 이러한 방식들은 상용 벤더에서 사용되고 있는데 Venti와 같은 고정형 중복제거(Fixed-size Chunking) 방식은 EMC의 Centera[7]에서 쓰였으며 LBFS와 같은 가변형 중복제거(Variable-size Chunking) 방식은 데이터도메인(Data Domain)과 Riverbed Technology사에서 쓰이고 있다.

클라이언트와 서버 구조와 순서에 따라서 중복 제거 모델을 나눌 수 있는데 클라이언트 기반, 인라인(in-line)[1], 포스트 프로세스(post-process)[2][3] 등 세 가지 모델이 있다. 클라이언트 측에서 파일 스트림을 해싱하고 서버에서 해싱 리스트의 중복을 확인하고 중복 되지 않은 블록들만을 서버로 전송하는 방식이다. 인라인 방식은 클라이언트에서 파일이 전송되는 순간에 해싱 작업과 해시 테이블 검색을 하여 중복 제거를 처리하는 방식이며 포스트 프로세스 방식은 파일이 전송되면 우선 임시 저장 장소에 파일을 저장하고 지정된 시간이나 시스템의 유휴 시간(idle time)에 저장되어 있는 파일을 읽어낸 후에 중복 제거 작업을 수행한다.

3. 시스템 설계 및 구현



[그림 1] 동적 모드 변경을 통한 QoS 보장 방식

본 연구에서는 각 방식의 장점을 취하여 시스템이 과부하인 상황에서도 지속적으로 중복 제거 작업을 수행할 수 있는 방안을 제시한다.

제안하는 방식은 클라이언트와 서버의 부하를 지속적으로 모니터링 하여 서버의 부하가 일정한 수준 이하인 경우에 인라인 방식으로 처리를 하고, 서버의 부하가 높은 경우에는 클라이언트 부하에 따라 클라이언트 기반 모델과 포스트 프로세스 모델로 변경한다. 만약 클라이언트의 부하가 낮은 경우에는 계산 량이 높은 해싱작업을 클라이언트에서 처리를 하는 방법을 사용하고 클라이언트와 서버의 부하가 높은 경우에 포스트 프로세스 모드로 전환하여 서버의 부하가 감소하는 시점에서 중복 제거 작업을 몰아서 수행한다.

사용자 관점에서는 서버의 부하가 높은 경우에 파일 전송이 늦어지는 문제를 클라이언트의 프로세서를 활용하여 해결을 할 수 있기 때문에 부하 분배가 클라이언트와 서버에서 균등하게 이루어지는 효과를 줄 수 있다. 또한 사용자가 QoS(Quality of Service)를 명세를 할 때 높은 등급의 사용자는 서버에서 대부분의 작업이 처리될 수 있게 허가를 하고, 낮은 등급의 사용자는 클라이언트 측에서 대부분의 작업이 처리가 되게 하여 사용자 등급에 따라서 리소스의 처리 방식에 변화를 준다.

■ 모드 조정(Mode Control)

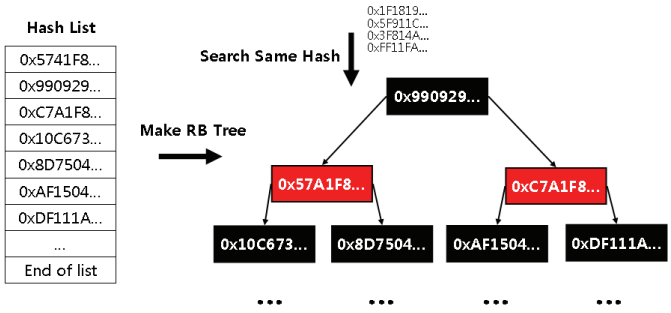
동적으로 모드를 조정하기 위해서 핵심 모듈이 클라이언트와 서버에 구현되어 있어야 한다. 핵심 모듈로는 CPU 자원을 많이 필요로 하는 해싱을 처리하는 모듈과, 해시를 비교하여 중복 제거 하는 모듈이 있다. 모드 컨트롤은 CPU 소모, 네트워크 현황과 같은 서버의 상태를 지속적으로 감시하고 인라인 방식, 포스트 프로세스, 클라이언트 기반 방법으로 변경할지 결정하지만 서버에 접속해 있는 모든 사용자가 같은 모드를 이행하는 것이 아니며 QoS 명세를 통해 모드와 작업 우선순위를 결정한다.

■ I/O

I/O 모듈에서 중점적으로 고려해야 할 사항으로 시스템 파일 캐쉬의 효율적인 이용이다. 램과 디스크의 처리 속도의 차이는 일반적으로 편차가 심하기 때문에 디스크를 여러 번 읽는 것을 경계해야 한다. 클라이언트 기반(Client-Based) 모드일 경우 클라이언트에서 파일을 읽고 해싱을 하고 중복 제거를 한다. 이 때 파일들의 검사를 한꺼번에 처리한 다음 해싱 작업을 수행할 경우 파일 캐쉬를 이용하지 못하고 디스크 I/O를 반복적으로 하게 된다. 또 포스

트 프로세스 모드의 서버 I/O에서도 클라이언트로부터 파일을 전송받고 해싱 작업을 할 경우 파일 캐쉬를 이용하지 못하고 디스크 작업을 하므로 수행시간이 늘어날 수 있다. 이런 문제들을 피하기 위해 파일 캐쉬를 추가적으로 늘리고 가용 가능한 파일 캐쉬 크기만큼 파일을 처리하는 기능을 삽입하여 성능을 늘렸다.

■ 해시 탐색(Hash Lookup)



[그림 2] RB Tree를 이용한 해시 데이터 관리

중복 제거 과정에서 해시를 비교하기 위해 효율적으로 해시를 저장하고 찾을 수 있는 방법이 필요하다. 본 논문에서는 중복 제거 과정 전에 데이터베이스의 해시를 각각의 레드블랙(Red-Black:RB) 트리에 로드하고 RB 트리를 통해 해시를 검색한다. 간단하게 데이터베이스를 사용하여 해시를 비교할 수 있지만 디스크를 접근하는 오버헤드가 생기기 때문에 메모리에서 처리할 수 있도록 자료 구조를 사용하고 있다. RB 트리는 빈번한 데이터 입력과 비교에 매우 효과적이다.

■ 해싱(Hashing)

다음 표는 본 연구에서 분석한 해시 수행 성능에 대한 결과이다. 데이터 크기에 따라서 성능이 변하는지에 대해서 실험을 수행하였다. 라빈(Rabin) 해시의 성능이 낮게 나오고 있지만 연속해시 이기 때문에 세밀하게 중복을 찾아내는 성능은 라빈 해시가 더 우위에 있다. 그러나 비트크기(Bit Lenth)가 낮기 때문에 비트크기가 큰 고정 형 해시들 보다 해시 충돌이 일어날 가능성이 높다.

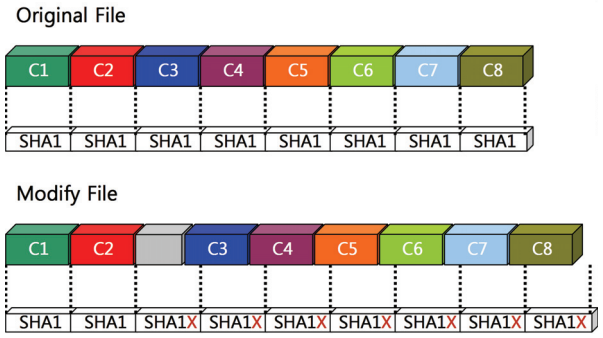
[표 2] 해시 수행 성능 (Mbps)

	Bit Lenth	1MB	50MB	100MB
SHA1	160 bits	608.7	922.7	905.9
RMD160	160 bits	304.2	306.1	301.9
MD5	128 bits	1258.1	1488.9	1484.7
Rabin	64 bits	180.4	189.3	198.2

■ 중복제거(De-duplication)

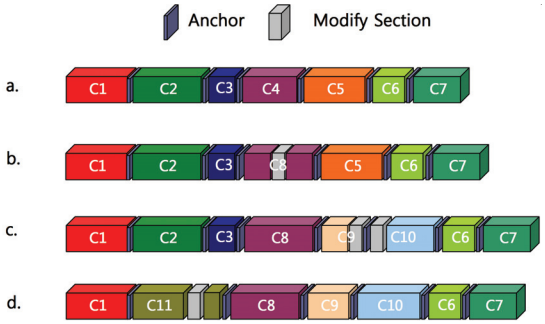
중복 제거 기술의 핵심 모듈인 De-duplication 모듈은 여러 중복 제거 알고리즘으로 구성한다. 큰 분류로는 고정 형 중복 제거와[4] 가변 형 중복 제거 방식(Variable Chunking)[5] 으로 나눈다.

DBC_FS(Duplicate Block Check with Fixed-size Blocking) 알고리즘은 파일을 일정한 크기의 블록으로 나누어 SHA1, MD5와 같은 해시함수를 적용하여 블록을 대표하는 해시를 생성한다. 생성된 해시를 기존의 스토리지에 저장되었던 해시 리스트와 비교하여 중복된 해시가 발견되면 연관된 블록도 중복 처리하는 것이다. DBC_FS 기법은 다른 종류의 중복제거 알고리즘보다 블록을 나누는 기법이 간단하기 때문에 빠른 수행 성능을 보여준다.



[그림 3] 고정 블록 방식 중복 제거 기법

CDC(Content-Define Chunking)은 파일의 삽입, 수정, 삭제 과정에서도 데이터의 경계를 유지하는 것이 특징이다. 데이터의 경계를 유지하는 것은 앵커(Anchor)라고 불리는 조그만 블록인데 이는 라빈 해시를 통해 나타나는 특정 값을 통해 확인된다. [그림 4]는 데이터 수정에 따른 CDC 알고리즘 적용 예시이다. CDC에서는 경계의 크기의 제한을 두어 데이터 수정 시 앵커의 간격이 너무 작거나 크지 않도록 조정하고 관리한다.

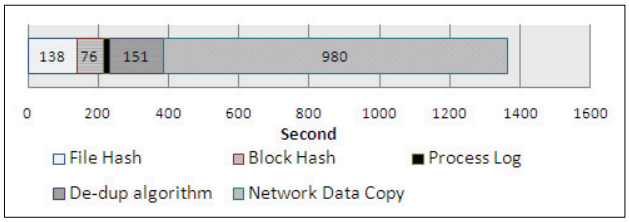


[그림 4] 가변 블록 기반 중복 제거 기법

■ 클라이언트 기반 및 서버 기반 성능 측정

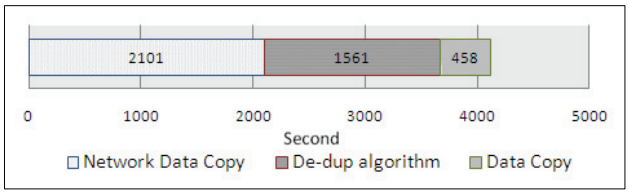
동적 모드변경 중복제거 서버 구현에 앞서 클라이언트 기반 모델과 서버 기반 모델을 구현하여 간단한 성능 측정을 하였다. 실험 환경은 Pentium 4 3.0GHz CPU, 웨스턴 디지털 WD-1600JS DISK, Memory 512MB, OS는 linux kerenl 2.6.18 이다.

[그림 5]는 클라이언트 기반 중복 제거 기술 모델의 구간별 수행 시간 측정 결과이며, 임의의(약 2Gbyte) 파일들을 중복 제거 실험한 것이다. 데이터 전송 시간이 모듈 수행 시간 중에서 가장 길며 다른 모듈의 수행 시간이 비교적 적은 이유는 클라이언트에서 주요 계산을 하고 서버로 넘겨주어 서버 CPU 소모를 줄이기 때문이다.



[그림 5] 클라이언트 기반 수행 시간 측정

[그림 6]은 서버 기반 중복 제거 기술 모델의 구간별 수행 시간 측정 결과이며, CentOS[7] 5.2 CD iso, DVD iso (약 7Gbyte) 파일들을 중복 제거 실험한 것이다. 중복 제거 모듈 수행 시간이 클라이언트 기반 수행 시간과 비교해 봤을 때 늘어난 것을 확인할 수 있다.



[그림 6] 서버 기반 모델 수행 시간 측정

네트워크 데이터 복사(Network Data Copy) 구간은 순수한 파일 복사 시간이며 데이터 복사는 중복이 제거된 데이터를 디스크 내에 이동시킬 때 소모되는 시간이다.

4. 결 론

본 논문은 중복 제거 시스템을 이용하는데 있어서 동적으로 모델을 변경하여 효율적으로 이용할 수 있는 중복 제거 시스템 제안에 대해 기술 하였다. 주요 기술 내용으로는 클라이언트와 서버의 부하를 모니터링하고 시스템 부하에 따라 모듈을 동적으로 변경할 때의 필요한 구성 요소를 살펴보고 동적 모드 변경은 아니지만 클라이언트 기반 모델, 서버 기반 모델을 구현하여 수행 시간 측정을 하였다. 실험 결과 클라이언트 기반 모델에서는 네트워크 데이터 전송이 더 필요하였지만 계산 시간을 줄일 수 있었다. 서버 기반 모델에서는 계산 시간이 늘어나는 점을 발견할 수 있었다. 이는 두 모델간의 장단점이 확연히 들어나는 것으로 서버의 부하에 따라 모드를 변경하여 이점을 얻을 수 있을 것임을 알 수 있다.

향후 연구로는 제안된 기술을 구현하여 기존의 기술의 성능을 향상 시키고 사용자 별로 QoS를 만족할 수 있도록 모듈을 설계할 것이다. 또한 대규모의 접속을 처리하고 확장성을 높일 수 있도록 클러스터링 기술을 접목할 계획이다.

참고문헌

[1] Data Domain, Data Domain Appliance Series: High-Speed Inline Deduplication Storage, 2005, http://www.datadomain.com/products/appliances.html

[2] LILLIBRIDGE,M., ESHGHI, K. Sparse indexing: large scale, inline deduplication using sampling and locality. In Proceedings of the 7th conference on File and storage technologies (2009), USENIX Association, pp. 111 - 123.

[3] "Hyperfactor: A breakthrough in Data Reduction Technology", Diligent (IBM) 2007

[4] QUINLAN, S., AND DORWARD, S. "Venti: a new approach to archival storage," In Proceedings of the 1st USENIX Conference on File and Storage Technologies (FAST), 2002.

[5] Athicha Muthitacharoen, Benjie Chen, and David Mazieres. A Low-Bandwidth Network File System. In Proceedings of the Symposium on Operating Systems Principles (SOSP'01), pages 174 - 187, 2001.

[6] RFC 3174, "US Secure Hash Algorithm 1 (SHA-1)"

[7] EMC CORPORATION. EMC Centera: Content Addressed Storage System, Data sheet, April 2002.

[8] centos home page, http://www.centos.org/