병렬 파일시스템을 위한 버저닝 기능 구현

차광호
한국과학기술정보연구원 슈퍼컴퓨팅센터
e-mail : khocha@kisti.re.kr

Implementation of the function of versioning on parallel file system

Kwangho Cha
Supercomputing Center, Korea Institute of Science and Technology Information

요 약

많은 파일 시스템은 각 구성 요소로부터 야기되는 다양한 오류에 노출되어 있어서 다양한 복구 기법이 제시되고 있다. 그러나 사용자의 실수로 인한 데이터 손실에 대해서는 상대적으로 대비책이 미비하다. 파일시스템 버저닝(virsioning)은 사용자의 부주의로 인한 데이터 손실을 최소화 할 수 있는 기술이다. 본 연구에서는 단일 시스템을 대상으로 개발된 버저닝 파일시스템인 ext3cow을 병렬 파일시스템인 PVFS2에 적용하여 병렬 파일시스템에서의 버저닝 기술의 적용 가능성을 살펴보았다.

1. 서론

파일 시스템의 성능을 개선하고자 하는 노력은 네트워킹 기법을 이용하여 다수의 디스크 내지는 스토리지를 연결하고 I/O처리를 분산시키는 병렬 파일 시스템의 개념을 만들어 내었다. 즉, 다수의 컴퓨터에 장착된 디스크나 스토리지를 네트워크로 연결하여 하나의 논리적인 파일 시스템으로 구성함으로써 유휴 자원의 활용, I/O처리 대역폭 증대 등의 효과를 기대할 수 있다. 특히 PVFS(Parallel Virtual File System)의 경우 클러스터 시스템의 확산과 함께 병렬 파일시스템 연구에 활발히 이용되고 있다[1,2].

파일시스템 버저닝(virsioning)은 사용자로 하여금 파일 시스템에 대하여 스냅샷들을 설정하고 미래에 이러한 스냅샷들에 접근할 수 있도록 하기 때문에 사용자의 부주의로 인한 데이터 손실을 최소화 할 수 있는 기법이다. 특히 리눅스 환경에서 단일시스템을 대상으로 개발된 ext3cow의 경우, 저널링 기능을 바탕으로 리눅스 시스템을 위한 버저닝 파일시스템을 제공하고 있다[3,4].

본 연구에서는 별도의 간단한 부수적인 프로그램을 추가하여 PVFS2에서 ext3cow를 기반으로 버저닝 기능을 이용할 수 있는 환경을 구축하고 이로 인한 오버헤드를 측정하였다.

2. 관련연구

2.1. PVFS2 (Parallel Virtual File System ver. 2)

파일 시스템의 성능을 높이기 위하여 많은 병렬 파일 시스템이 취하는 방식이 RAID 0처럼 파일을 쪼개서(stripe) 서로 다른 볼륨에 저장하는 것이다. Clemson 대학에서 개발된 PVFS(Parallel Virtual File System) 역시 I/O를 담당하는 복수 I/O노드에 파일이 분산되어 저장되며 이에 대한 위치 정보를 관리하는 관리 노드가 별도로 존재한다. 이때 I/O노드와 관리 노드의 역할을 하는 프로세스는 단일 노드 내에 위치 할 수 있다.

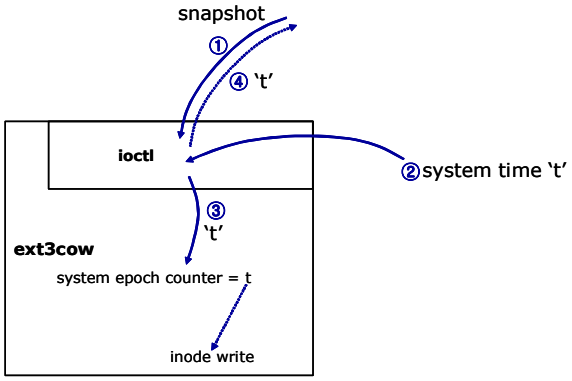
PVFS의 기능에 설치의 편리성, 이질적인 클러스터 시스템 지원 및 스토리지 및 네트워크를 위한 모듈화 기능 강화 등을 고려하여 추가 개발된 것이 PVFS 버전 2이다. PVFS가 TCP/IP위주의 프로토콜을 사용하는 반면, PVFS2는 클러스터 시스템용 고성능 네트워크인 미리넷과 인피니밴드를 위한 프로토콜의 지원도 포함하고 있다[1,2].

2.2. 버저닝(Virsioning) 파일시스템

기능과 구현방법에 따라 다양한 버저닝 파일시스템이 존재하는데 Cedar [5]는 가장 초기의 버저닝 파일시스템의 한 예이다. 시간대에 따른 파일별 버전을 생성하며 매 쓰기 연산마다 새로운 버전을 생성하게 된다. 앤드류 파일 시스템[6]의 경우 copy-on-write 방식을 이용한 스냅샷을 지원하는데 이는 온라인 환경에서 신속하고 오버헤드가 적은 실행에 초점을 두고 있다. 확장성을 고려하여 사용자 레벨에서 구현된 Wayback [7]은 각각의 쓰기 요청을 undo log에 저장하고 이전 버전으로 rollback시 사용하도록 하였다. Ext3cow의 경우 각 파일에 대한 시간 변경 기능을 제공하고 있으며 ext3를 기반으로 개발되었다. Ext3에 버저닝 기능을 포함하기 위하여 inode에 수정을 가하여 시간 정보 및 copy-on-write관련 정보를 유지하고 있다[3,4]. 이외에도 다양한 기법을 이용한 버저닝 파일시스템을 찾아 볼 수 있다[8,9].

3. PVFS를 위한 버저닝 환경

버저닝 파일시스템의 가장 핵심적인 기능은 스냅샷을 설정하는 것이다. 본 연구에서 사용된 ext3cow의 경우 그림 1과 같이 ext3cow의 시간변수를 새로운 시간으로 설정하는 것이 주요 내용이다.



(그림 1) ext3cow의 snapshot절차: ① 사용자가 스냅샷을 수행한다. ②~③ ext3cow의 ioctl을 통하여 시스템 시간을 구하고 이를 시간변수로 설정한다. 이후 파일이 기록되거나 수정될 때 이 시간변수를 이용한다. ④ 사용된 시스템 시간이 사용자에게 반환된다.

이처럼 ext3cow의 스냅샷은 단일 시스템에서 수행되는 것을 조건으로 하기 때문에 다수의 IO서버를 사용하는 PVFS2에서는 이 과정이 수정되어야만 한다. 즉 각각의 IO서버가 서로 다른 시스템 시간을 사용하는 상황에서 동시에 스냅샷을 수행하고 이를 시간대 변경(time-shift) 시에 사용하기 위해서는 공통된 시간 정보가 필요하였고 이를 위하여 전역 시간(global time)을 이용한 스냅샷을 설계하였다.

그림 2는 본 연구에서 구현한 버저닝 PVFS2의 구조와 그 수행 과정을 보여준다. 각 과정은 다음과 같다.

① PVFS2 사용자로부터 스냅샷 명령이 실행된다.

② PVFS2의 management 서버에 존재하는snapshot 프로그램은 각 IO 서버에게 지역시간 정보를 요청하며, 각 IO서버는 이에 대한 응답으로 자신들의 시스템 시간을 management 서버에게 전달한다.

③ Management 서버는 각 IO서버의 지역 시간을 바탕으로 전역 시간(global time)을 설정한 뒤 각 IO서버에 전달한다.

④~⑤ 각 IO서버는 ext3cow와 같은 방식으로 전역시간을 이용하여 스냅샷을 수행한다.

⑥ 스냅샷이 성공한 시간 정보를 기록하여 시간 변경에 사용할 수 있도록 한다.

4. 구현

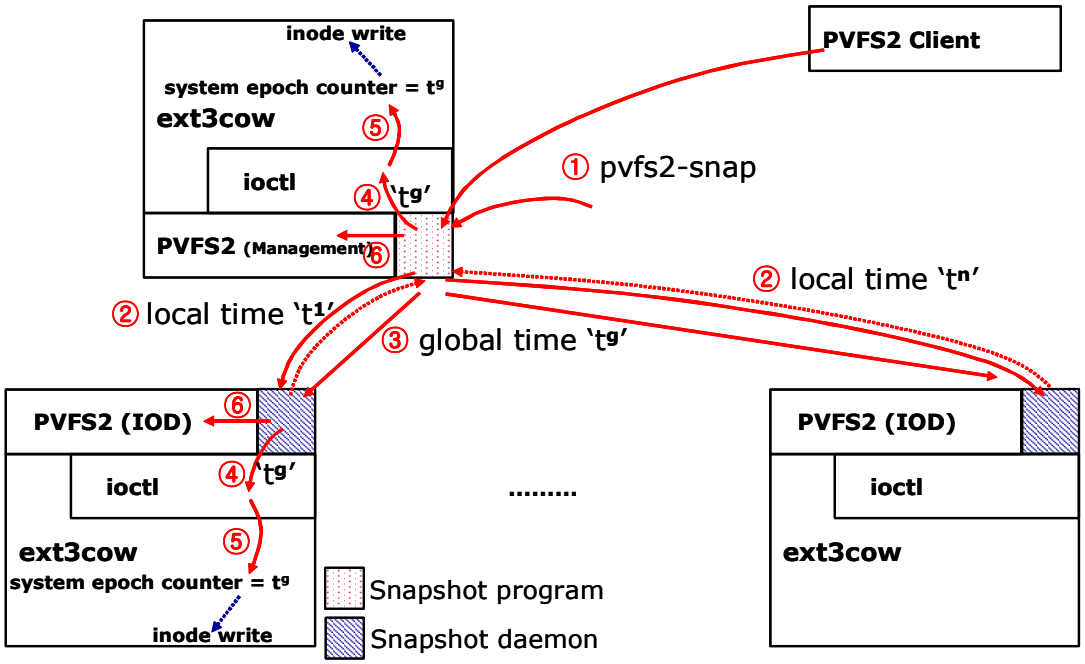
본 장에서는 구현 환경과 실험 결과에 대하여 간략하게 설명한다.

4.1. 구현 환경

표 1은 구현에 사용된 소규모 클러스터 시스템의 하드웨어 시스템 구성으로 보여주며 표 2는 소프트웨어 구성을 보여준다.

<표 1> 하드웨어 구성 요소

1 Client, 1 Management node, and 3 IODs	
CPU	Intel Pentium 4 XEON 2.8
Memory	1GB
HDD	80GB SATA
Networks	Fast Ethernet, Gigabit Ethernet



(그림 2) PVFS를 위한 버저닝 파일 시스템 구조 및 수행 절차

PVFS Client	PVFS Management server
[khocha@c01 pvfs2]$ ls ./test/ -l total 3324 -rw-r--r-- 1 khocha khocha 243495 Dec 15 12:22 pvfs-1.6.3.tgz -rw-r--r-- 1 khocha khocha 3155014 Dec 15 12:22 pvfs2-1.3.0.tar.gz [khocha@c01 pvfs2]$ date >> log.txt [khocha@c01 pvfs2]$ cat log.txt Thu Dec 15 12:23:48 GMT 2005 [khocha@c01 pvfs2]$ /usr/local/bin/pvfs2-snap Snap_shot: 1134649458	
[khocha@c01 pvfs2]$ rm ./test/pvfs-1.6.3.tgz [khocha@c01 pvfs2]$ ls ./test/ -l total 3084 -rw-r--r-- 1 khocha khocha 3155014 Dec 15 12:22 pvfs2-1.3.0.tar.gz [khocha@c01 pvfs2]$ date >> log.txt [khocha@c01 pvfs2]$ cat log.txt Thu Dec 15 12:23:48 GMT 2005 Thu Dec 15 12:25:37 GMT 2005	
	[root@compute-0-6 ver01]# ./mklink.sh 1134649458 [root@compute-0-6 ver01]# ./repvfs.sh restart Stopping PVFS2 server: Starting PVFS2 server: [OK] Stopping PVFS2 server: Starting PVFS2 server: [OK] Stopping PVFS2 server: Starting PVFS2 server: [OK] Stopping PVFS2 server: Starting PVFS2 server: [OK]
[khocha@c01 pvfs2]$ cat log.txt Thu Dec 15 12:23:48 GMT 2005 [khocha@c01 pvfs2]$ ls ./test/ -l total 3324 -rw-r--r-- 1 khocha khocha 243495 Dec 15 12:22 pvfs-1.6.3.tgz -rw-r--r-- 1 khocha khocha 3155014 Dec 15 12:22 pvfs2-1.3.0.tar.gz	

(그림 3) PVFS를 위한 버저닝 파일 시스템의 운영 테스트 결과

<표 2> 소프트웨어 구성 요소

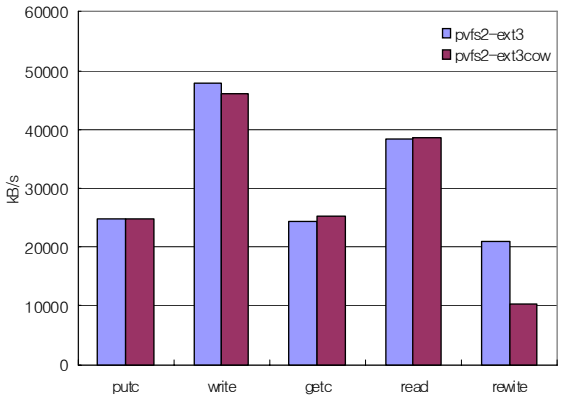
OS	Linux - 2.4.21
PVFS	PVFS2 - 1.3.0
Versioning file system	ext3cow - 0.1.3

일반적인 네트워크를 사용하는 클러스터 시스템을 기본으로 하며 초기 ext3cow를 사용하여 테스트 환경을 구축하였다.

4.2. 운영 실험

그림 3은 임의의 시간에 대한 스냅샷의 생성과 특정 시간으로 파일 시스템의 상태를 변경하는 과정을 보여 주고 있다. 클라이언트 측에서 파일의 정상적인 생성 여부를 확인하고 스냅샷을 생성하였다. 특정 스냅샷 시간으로 이동(time-shift)하기 위해서는 원하는 스냅샷 시간을 기술한 뒤 전체 PVFS 서버를 재가동하는 방식을 취하고 있다.

스냅샷 기능을 추가하면서 기존의 PVFS에 어느 정도 성능 저하가 나타나지를 확인할 필요가 있었다. 이를 확인하기 위하여 파일시스템 성능 측정에 많이 사용되는 bonnie 테스트를 수행하였다.



(그림 4) Bonnie 테스트 결과

그림 4은 기존의 ext3를 이용하는 PVFS와 ext3cow를 기반으로 구축한 버저닝 기능을 부가한 PVFS의 IO 성능을 보여 주고 있다.

블록 IO테스트의 경우 write에서 일부 성능저하가 발견되었고 특히 rewrite의 경우 성능저하를 보이고 있는데 이는 사용된 초기 ext3cow의 특성으로부터 야기된 결과로 예측된다.

5. 결론

PVFS는 병렬파일 연구에 가장 많이 이용되는 파일 시스템이라 할 수 있다. 최근 많은 연구가 PVFS자체의 성능에 초점이 맞추어졌으나, 운명 및 관리의 측면에서는 그 외 부수적인 기능도 고려되어야 한다.

본 연구에서는 PVFS에 ext3cow의 기능을 사용할 수 있는 부가적인 프로그램들을 추가하여 PVFS에서 버져닝 기능을 수행할 수 있도록 하였다. ext3cow의 특성으로 인한 어느 정도의 성능 저하는 발견되었으나 버져닝 기능의 사용에는 무리가 없었다.

참고문헌

[1] Parallel Virtual File System 2, http://www.pvfs.org/pvfs2.

[2] P. H. Carns, W. B. Ligon III, R. Ross, and P. Wyckoff, "BMI: a network abstraction layer for parallel I/O," Workshop on Communication Architecture for Clusters, Proceedings of IPDPS '05, 213a, 2005.

[3] Z. Peterson, and R. Burns, "Ext3cow: A Time-Shifting File System for Regulatory Compliance," In ACM Transactions on Storage, 1(2):190~212, May, 2005.

[4] Z.N.J. Peterson, and R.C. Burns, "Ext3cow: The Design, Implementation," and Analysis of Meta-data for a Time-Shifting File System, Technical Report HSSL-2003-03, Department of Computer Science, Johns Hopkins University, 2003.

[5] D. K. Gifford, R.M. Needham, and M.D. Schroeder, "The Cedar File System," Communication of the ACM, 31(3):288~298, 1988.

[6] J. Howard, M. Kazar, S. Menees, D. Nichols, M. Satyanarayanan, R. Sidebotham, and M. West, "Scale and Performance in a Distributed File System," ACM Transactions on Computer Systems, 6(1):51~81, 1988.

[7] B. Cornell, P. A. Dinda, and F. E. Bustamante, "Wayback: A User-level Versioning File System for Linux," In Proceedings of the USENIX Annual Technical Conference, 19~28, 2004.

[8] S. Quinlan, and S. Dorward, "Venti: A new approach to archival storage," In Proceedings of the 2002 Conference on File And Storage Technologies (FAST), 89~101, 2002.

[9] M. Rosenblum and J. Ousterhout, "The Design and Implementation of a Log-Structured File System," ACM Transactions on Computer Systems, 10(1):26~52, 1992.