캔 클러스터 파일 시스템의 설계 및 구현

황인철° 임동혁 김호진 맹승렬 조정완
한국과학기술원 전자전산학과 전산학전공
{ichwang°, dhlim, hojin, maeng, jwcho}@calab.kaist.ac.kr

Design and Implementation of CAN Cluster File System

In-chul Hwang° Donghyouk Lim Hojin Ghim Seung-Ryoul Maeng Jung-Wan Cho
Division of Computer Science, Dept. of Electrical Engineering & Computer Science, KAIST

요 약

요즘 네트웍과 PC의 성능이 향상됨에 따라 값싼 PC를 빠른 네트웍으로 묶어 높은 성능을 얻고자 하는 클러스터 시스템에 대하여 많이 연구 되어 왔다. 이러한 연구의 한 분야로서 클러스터 시스템에서 각 노드의 CPU나 메모리에 비하여 상대적으로 느린 디스크에 접근하는 파일 시스템을 효율적으로 구성하려는 연구가 이루어지고 있다.

기존 클러스터 파일 시스템은 기존에 연구되었던 분산 시스템의 파일 시스템을 그대로 사용하는 경우가 많았다. 기존 분산 시스템들은 클러스터 시스템과 유사한 부분들이 존재 하지만 다른 부분도 존재한다. 클러스터 시스템을 사용하는 사용자에게 높은 성능의 데이터 입출력과 효율적인 지원을 위해서는 클러스터 시스템의 특성을 잘 활용하는 클러스터 파일 시스템에 대한 연구가 필요하다.

본 논문에서는 클러스터 시스템의 특성을 잘 활용하는 캔 클러스터 파일 시스템의 설계 및 구현에 대하여 기술한다. 캔 클러스터 파일 시스템은 자료 저장 시스템을 클러스터 시스템의 특성을 잘 활용하는 단일 디스크 입출력을 사용하고 그 위에 상호 협력 캐쉬를 구현함으로서 높은 대역폭의 데이터 입출력을 제공한다. 이러한 캔 클러스터 파일 시스템의 성능을 기존 파일 시스템 중 PVFS와 테스트 프로그램 수행을 통하여 성능을 비교, 분석한다.

1. 서론

요즘 네트웍과 PC의 성능이 향상됨에 따라 값싼 PC를 빠른 네트웍으로 묶어 높은 성능을 얻고자 하는 클러스터 시스템에 대하여 많이 연구 되어 왔다. 이러한 연구의 한 분야로서 클러스터 시스템에서 각 노드의 CPU나 메모리에 비하여 상대적으로 느린 디스크에 접근하는 파일 시스템을 효율적으로 구성하려는 연구가 이루어지고 있다.

기존 클러스터 파일 시스템은 기존에 연구되었던 분산 시스템의 파일 시스템을 그대로 사용하는 경우가 많았다. 기존 분산 시스템들은 클러스터 시스템과 유사한 부분들이 존재 하지만 다른 부분도 존재한다. 클러스터 시스템을 사용하는 사용자에게 높은 성능의 데이터 입출력과 효율적인 지원을 위해서는 클러스터 시스템의 특성을 잘 활용하는 클러스터 파일 시스템에 대한 연구가 필요하다.

본 논문에서는 클러스터 시스템의 특성을 잘 활용하는 캔 클러스터 파일 시스템의 설계 및 구현에 대하여 기술한다. 캔 클러스터 파일 시스템은 기존 분산 파일 시스템들에서 제시된 캐쉬의 일관성을 이용하여 캐쉬의 일관성을 보장 시켜준다. 하지만 기존 분산 파일 시스템과 달리 자료 저장 시스템을 각 노드에 있는 디스크를 효율적을 묶는 단일 디스크 입출력 모듈을 이용함으로서 기존 분산 파일 시스템과 다른 캐쉬 관리나 블록의 활용을 이용하게 된다. 또한 분산된 환경에서의 관리 부하를 줄이기 위해서 효율적인 데이터 할당과 관리를 하도록 구현되었다. 그리고 상호 협력 캐쉬를 사용함으로서 여러 노드의 캐쉬를 효율적으로 공유하여 사용자에게 높은 대역폭의 데이터 입출력을 제공한다. 이러한 캔 클러스터 파일 시스템의 성능을 기존 분산 파일 시스템 중 PVFS와 테스트 프로그램 수행을 통하여 성능을 비교, 분석한다.

본 논문의 구성은 다음과 같다. 2장에서는 관련 연구로서 기존의 분산 파일 시스템들과 상호 협력 캐쉬에 대하여 살펴본다. 3장에서는 캔 클러스터 파일 시스템의 설계와 구현에 대하여 설명한다. 4장에서는 캔 클러스터 파일 시스템과 PVFS의 성능을 측정하고 비교 분석한다. 5장에서는 향후 연구 방향과 결론을 맺는다.

2. 관련 연구

2.1 분산 파일 시스템

기존 분산 시스템에서 효율적인 데이터 활용을 위하여 분산 파일 시스템에 대한 연구가 활발히 이루어 졌다. 기존 분산 파일 시스템으로 NFS[1], AFS[2], PVFS[3], GFS[4] 등이 개발되었다. 서버-클라이언트 구조의 분산 파일 시스템인 NFS나 AFS, Coda, PVFS는 서버에 저장된 데이터를 공유하기 위해 개발된 것으로 서버의 디스크만 활용할 뿐 클러스터 시스템의 분산된 디스크를 효율적으로 사용할 수 없다는 단점이 있다. GFS는 특정 하드웨어인 SAN(Storage Area Network)상에서 개발된 파일 시스템으로 고가의 하드웨어 장치를 요구한다는 단점이 있다.

클러스터 내부의 분산된 디스크를 효율적으로 활용할 수 있는 파일 시스템으로 Frangipani[5]가 개발되었다. Frangipani는 분산된 디스크를 하나의 디스크 풀로 만들어 주는 Petal[6] 시스템 상에서 락과 파일 시스템 인터페이스의 구현으로 구현되었다. Frangipani는 캔 클러스터 파일 시스템과 유사한 구조를 지니고 있으나 클러스터의 특성을 활용하지 못한다는 단점이 있다.

2.2 상호 협력 캐쉬

상호협력 캐쉬[7,8,9]는 기존 요구 처리 단계 중 클라이언트가 자신의 요구가 자신의 캐쉬에서 처리가 되지 않을 경우 서버에게 요청하기 전 그 블록을 캐싱하고 있는 다른 클라이언트에게 그 블록에 대한 요청을 하여 자신의 요구를 처리하는 방

법이다.

이러한 상호협력 캐쉬에 대하여 많은 연구가 이루어 졌다. Dahlin[7]은 캐쉬 블록들의 관리를 위하여 N-chance 알고리즘을 제안하였고, Feeley[8]는 GMS(Global Memroy Service) 상에서 효율적인 캐쉬 블록 알고리즘을 제안하였다. 그리고 Sarkar[9]는 기존 상호협력 캐쉬에서 정확한 클라이언트 캐싱 정보를 가지고 있던 것을 단순한 힌트에 의해 블록의 캐싱 정보를 유지함으로서 캐싱 정보를 유지하는데 필요한 부하를 줄이는 방법을 제안하였다.

캔 클러스터 파일 시스템에서는 이러한 상호 협력 캐쉬를 사용함으로서 높은 대역폭의 데이터 입출력을 제공하여 준다.

3. 캔 클러스터 파일 시스템의 설계 및 구현

3.1 Overview

다음 그림 1은 캔 클러스터 파일 시스템의 구조를 보여준다.

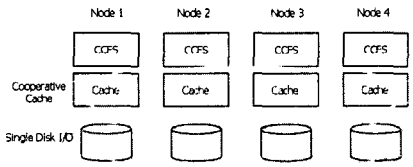

그림 1. 캔 클러스터 파일 시스템의 구조

캔 클러스터 파일 시스템은 각 노드의 디스크를 효율적으로 하나의 모습으로 보여주는 단일 디스크 입출력 시스템[10]을 기반으로 구성되며 파일 시스템과 단일 디스크 입출력을 위한 커널 모듈 사이에는 상호 협력 캐쉬가 캐싱과 버퍼링을 수행함으로서 높은 성능의 데이터 입출력을 제공한다.

3.2 캔 클러스터 파일 시스템의 데이터 배열 및 관리

다음 그림 2는 캔 클러스터 파일 시스템의 데이터 배열을 보여준다.

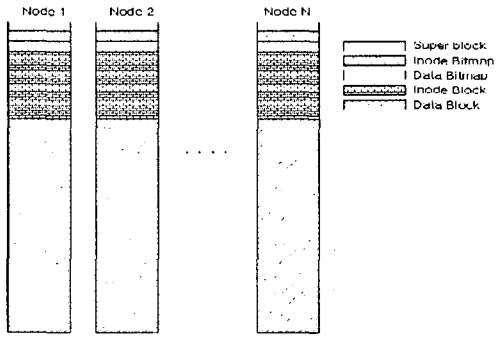

그림 2. 캔 클러스터 파일 시스템에서의 데이터 배열

그림 2에서와 같이 캔 클러스터 파일 시스템에서 각 노드별로 수퍼 블록(super block)을 저장하여 관리한다. 기존 분산 파일 시스템에서 수퍼 블록은 여러 노드에 공유되기 때문에 수퍼 블록에 대한 관리의 부하가 큰 단점이 있다. 하지만 캔 클러스터 파일 시스템에서는 각 노드별로 데이터 관리하도록 하여 수퍼 블럭이 공유되지 않고 각 노드별로 수퍼 블록을 관리함으로서 수퍼 블록 관리의 부하를 줄였다. 마찬가지로 캔 클러스터 파일 시스템에서는 여러 노드에 공유되어야 하는 아이노드 비트맵과 데이터 비트맵도 노드별로 비트맵을 관리하게 하여 비트맵에 대한 관리 부하를 줄였다. 캔 클러스터 파일 시스템에서 아이노드 블록의 할당은 자신의 노드에서 하며 데이터 블록의 할당은 여러 노드에 분산되어 할당을 시킴으로서 높은 대역폭을 제공할 수 있다.

3.2 캔 클러스터 파일 시스템에서의 상호 협력 캐쉬와 데이터 일관성 보장

캔 클러스터 파일 시스템에서 상호 협력 캐쉬는 단일 디스크 입출력 상에서 개발이 되었지만 자신의 노드에 있는 디스크로부터의 데이터 입출력 속도와 다른 노드의 디스크로부터의 데이터의 입출력 속도가 다르기 때문에 자신의 디스크의 데이터에 대한 흠을 자신으로 설정하고 관리하는 흠 기반 상호 협력 캐쉬로 구현하였다. 캔 클러스터 파일 시스템에서의 캐쉬를 흠 기반 상호 협력 캐쉬로 구현함으로서 자신의 캐쉬에 없는 데이터에 대한 요구를 단일 디스크 입출력에게 할 필요 없이 흠 노드의 캐쉬 관리자에게 요청함으로서 효율적인 데이터의 전송이 가능하게 된다.

캔 클러스터 파일 시스템에서의 상호 협력 캐쉬에서 노드간의 캐쉬 일관성은 메타데이터와 일반 파일 데이터에 따라 다르게 적용되었다. 메타데이터의 경우 여러 노드에서도 빈번히 사용되기 때문에 변경된 내용을 다른 노드의 캐쉬에 적용시키는 업데이트 기반 정책을 사용하였고, 데이터의 경우는 변경된 내용이 있으면 다른 노드에 캐싱된 내용을 해제시키는 해제 기반 정책을 사용하였다. 그리고 메타데이터와 데이터의 효율성을 높이기 위해서 메타데이터와 데이터의 크기를 다르게 하여 관리하는 방법을 사용하였다. 메타데이터의 경우 파일에 대한 정보만 유지하면 되기 때문에 메타데이터를 저장하는 블록의 크기가 데이터와 같이 클 필요가 없지만 데이터의 경우는 실제의 큰 데이터를 저장하기 때문에 높은 대역폭과 큰 파일을 지원하기 위해서 메타데이터보다 훨씬 큰 크기의 블록에 저장하도록 하였다.

4. 성능 평가

성능 평가를 위해 사용된 환경은 다음 표 1과 같다.

CPU	Pentium IV 1.8GHz
Memory	512MByte 266MHz DDR Memory
Disk	IBM 60G 7200rpm
Network	3c996B-T(Gigabit Ethernet) 3c17701-ME (24port Gigabit Ethernet Switch)
OS	RedHat 9.0(Kernel 2.4.20)
PVFS	Version 1.6.0

표 1. 성능 평가 환경

4노드를 사용하여 평가하였으며 PVFS는 I/O서버는 4노드에 메타데이터 관리자는 1개의 노드를 할당하여 실험을 수행하였다. 캔 클러스터 파일 시스템에서 메타데이터 블록의 크기는 1KB로 하였으며 데이터 블록의 크기는 PVFS의 스트라이핑 크기와 같이 64KB로 하여 실험을 수행하였다.

테스트 프로그램은 4096*4096 두개의 행렬을 파일로부터 읽고 곱셈을 수행하여 그 결과를 파일에 적는 응용프로그램과 512*512 행렬의 내용을 파일로부터 읽어 유체의 흐름을 시뮬레이션하는 Equation Solver Kernel[11] 프로그램을 수행시키고 그 결과를 파일에 적고 한 노드에서 그 결과를 파일로부터 읽어서 화면에 출력하는 프로그램을 20회 실행시켰을 경우 응용프로그램에서의 읽기/쓰기 시간을 측정하였다.

4.1 행렬 곱셈 프로그램 실행 결과

다음 그림 3은 행렬곱셈 프로그램 수행 시간 중 읽기/쓰기 시간을 나타낸다.

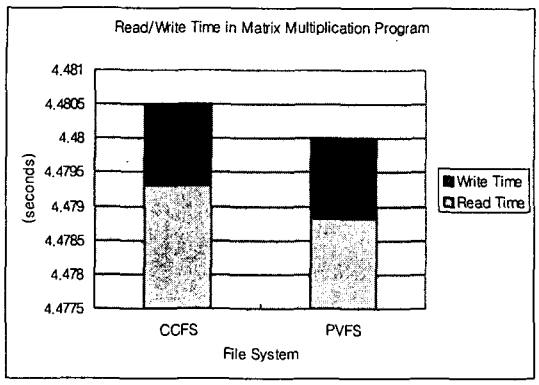

그림 3. 행렬 곱셈 프로그램 수행 시간 중 읽기/쓰기 시간

행렬 곱셈 프로그램에서는 입력을 받는 파일은 계속 읽기를 수행하고 출력을 하는 파일은 계속 쓰기를 수행한다. 행렬 곱셈 프로그램과 같이 파일에 대한 읽기/쓰기가 명확하게 이루어질 경우 그림 3에서와 같이 캔 클러스터 파일 시스템과 PVFS의 성능은 크게 차이나지 않는다.

4.2 Equation Solver Kernel 프로그램 실행 결과

다음 그림 4는 Equation Solver Kernel 프로그램 실행 시간 중 읽기/쓰기 시간을 나타낸다.

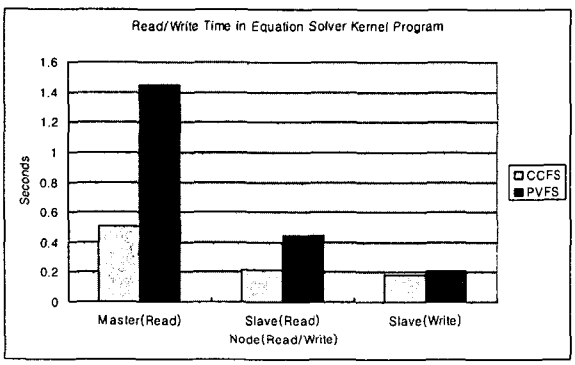

그림 4. Equation Solver Kernel 프로그램 수행 시간 중 읽기/쓰기 시간

Equation Solver Kernel 프로그램은 하나의 파일에 대하여 읽기/쓰기를 수행하게 된다. 이런 경우 그림 4에서 보는 바와 같이 읽기의 경우 캔 클러스터 파일시스템의 성능이 PVFS의 성능보다 약 50%정도 빠른 것을 알 수 있다. 캔 클러스터 파일 시스템은 단일 디스크 입출력을 이용하여 병렬 I/O를 이용할 뿐 아니라 홈 기반 상호 협력 캐쉬를 이용하여 효율적인 캐쉬을 수행하기 때문에 PVFS에 비해 더 빠른 읽기 성능을 나타낸다. 쓰기의 경우 캔 클러스터 파일 시스템이 PVFS에 비해 약 10%정도 더 빠른 성능을 나타낸다. 쓰기의 경우 캔 클러스터 파일 시스템은 다른 노드에 캐쉬되어있던 모든 내용을 해제시킨 후 홈 노드에서 버퍼링을 수행하는데 이 시간이 직접 I/O 서버에 데이터를 전송하는 PVFS보다 짧기 때문에 더 좋은 성능을 나타낸다.

5. 향후 연구 방향 및 결론

본 논문에서는 클러스터 시스템의 특성을 효율적으로 활용하여 높은 대역폭을 지원해 주는 캔 클러스터 파일 시스템의 설계 및 구현에 대하여 설명하였다. 캔 클러스터 파일 시스템은 여러 노드에 분산된 디스크를 효율적으로 묶는 단일 디스크 입출력 서비스를 자료 저장 시스템으로 사용할 뿐 아니라 파일 시스템 데이터를 분산된 디스크에 맞게 설계하여 효율적으로 데이터 관리를 수행한다. 또한 여러 분산된 노드의 캐쉬를 상호 협력 캐쉬로 묶어 데이터의 효율적인 공유를 가능하게 한다. 성능 평가 결과 기존 분산 파일 시스템 중 PVFS에 비교하였을 때 읽기/쓰기 시간은 0.2%~50% 정도의 성능 향상이 있었다.

앞으로 캔 클러스터 파일 시스템의 더 많은 성능 분석을 통하여 성능의 병목 현상을 일으키는 부분을 찾아 개선할 예정이며, 파일 시스템 자체의 데이터 관리를 효율적으로 할 수 있는 기법에 대해 연구할 예정이다. 또한 캔 클러스터 파일 시스템에서의 효율적인 캐쉬 관리 기법에 대하여 연구할 예정이다.

6. 참고 문헌

[1] "Linux NFS faq", http://nfs.sourceforge.net/
[2] AFS (Andrew File System)
http://www.angelfire.com/hi/plutonic/afs-faq.html
[3] PVFS(Parallel Virtual File System)
http://www.parl.clemson.edu/pvfs/
[4] GFS(Global File System)
http://www.sistina.com/gfs/
[5] Chandramohan A. Thekkath, Timothy Mann, Edward K. Lee, "Frangipani: A Scalable Distributed File System", Symposium on Operating Systems Principles, 1997.
[6] Edward K. Lee, Chandramohan A. Thekkath, "Petal: Distributed Virtual Disks", Proceedings of the Seventh International Conference on Architectural Support for Programming Languages and Operating Systems 1996.
[7] Dahlin, M., Wang, R., Anderson, T., and Patterson, D. 1994. "Cooperative Caching: Using remote client memory to improve file system performance", In Proceedings of the First USENIX Symposium on Operating Systems Design and Implemntation. USENIX Assoc., Berkeley, CA, 267-280
[8] Feeley, M. J., Morgan, W. E., Pighin, F. H., Karlin, A. R., and Levy, H. M. 1995. "Implementing global memory management in a workstation cluster", In Proceedings of the 15th symposium on Operating System Principles(SOSP). ACM Press, New york, NY, 201-212
[9] Prasenjit Sarkar , John Hartman, "Efficient cooperative caching using hints", Proceedings of the second USENIX symposium on Operating systems design and implementation, p.35-46, October 29-November 01, 1996, Seattle, Washington, United States
[10] 황인철, 김동환, 김호진, 맹승렬, 조정완, "단일 디스크 입출력을 위한 커널 모듈 프로토타입의 설계 및 구현",한국정보과학회 2003년도 추계학술발표논문집, 2003년 10월
[11] D. Culler and J. Singh, "Parallel Computer Architecture : A Hardware/Sofware Approach", Morgan Kaufmann Publishers Inc.,I