결함내성 원격 메모리 시스템에 관한 연구

정형수, 한 혁, 김신규, 염헌영
서울대학교 전기컴퓨터공학부
e-mail:{jhs, hhyuck, sgkim, yeom}@dcslab.snu.ac.kr

A Study on a Fault-tolerant Remote Memory System

Hyungsoo Jung, Hyuck Han, Shin Gyu Kim, Heon Y. Yeom
Dept of Computer Science & Engineering, Seoul Nationl University

요 약

본 연구에서는 대용량 메모리 데이터 처리를 위한 범용 하드웨어 기반의 결함내성 원격 메모리 시스템의 성능을 분석하고자 한다. 대단히 빠른 접근 속도를 보장하는 휘발성 메모리를 이용한 원격 메모리 시스템의 실용적인 활용을 위해서는 결함내성의 성질이 필수적으로 보장되어야 한다. 본 연구에서는 RAID 기법과 유사한 방법을 이용하여 결함내성 메모리 시스템을 구현하고, 제안한 새로운 계층의 결함내성 메모리의 성능을 평가하고자 한다. 범용으로 쓰이는 MySQL과 같은 데이터베이스를 이용한 TPC-C 실험 결과로 볼 때 본 연구에서 구현한 결함내성 원격 메모리 시스템은 일반적인 대용량 메모리 데이터 처리 시스템에서 요구하는 필수요건인 결함내성 성질을 성공적으로 만족하고 있는 것으로 생각된다.

1. 서론

디스크의 접근 속도보다 대단히 빠른 응답성능을 보장하는 휘발성 메모리(RAM)의 특징은 대용량 메모리 데이터 처리를 위한 원격 메모리 시스템의 활발한 연구로 이어져왔다. 하지만, 그러한 성능우위의 특징에도 불구하고 현재 대용량 데이터 처리 분야에서는 이러한 특징을 충분히 활용하고 있지 못한 실정이다. 주요한 원인은 대용량 데이터 처리에 있어서 분산된 환경에서도 안전하게 계산을 완료할 수 있도록 만들 수 있는 결함내성 성질의 효율적인 구현 문제가 검증되지 않았기 때문이다. 본 연구에서는 RAID 기법을 응용한 결함내성 원격 메모리 시스템을 구현하여 대용량 메모리 데이터 처리 프로그램에서 요구하는 실용적인 수준의 성능 분석을 하고자 한다.

2. 결함내성 원격 메모리 시스템 구조

메모리와 디스크의 접근 속도의 차이는 많은 연구자들로 하여금 원격 메모리 시스템에 관한 연구를 수행하게 하였고, 다양한 기법들이 유수의 학회지에 제출되어 그 가능성을 볼 수 있었다[1,2,3,4,5,6,7,8]. 원격 메모리 시스템의 근본적인 개념은 휘발성 메모리와 물리적 디스크 사이에 새로운 메모리 계층을 원격에 있는 컴퓨터의 메모리를 이용하여 생성하는 것을 기반으로 하고 있다. 그림 1에서 나타난바와 같이 원격 메모리는 빠른 휘발성 메모리와 느린 디스크 사이에서 의미 있는 응답시간과 용량을 제공하는 메모리 계층으로 나타내지고 있다.

본 연구에서 제안하고 있는 결함내성 원격 메모리 시스템은 운영체제 커널에 구현되어 있다. 메모리와 디스크 사이에 새로운 메모리 계층으로 역할을 수행하기 위해 기존 운영체제의 스왑(swap) 시스템을 수정하여 원격 메모리의 사용은 로컬 메모리가 부족한 상황에서만 발생할 수 있도록 하였다. 본 연구가 제안한 결함내성 원격 메모리 시스템은 두 가지의 큰 특징을 가지고 있다. 첫 번째로, 원격 컴퓨터에 있는 메모리의 안전한 공유와 효율적인 데이터 전송을 위하여 RDMA(Remote Direct Memory Access) 기능이 가능한 네트워크 인터페이스를 이용하여 원격 메모리의 페이지 전송을 처리하는 것이다. 두 번째로, 원격 메모리를 제공하는 컴퓨터의 결함 상황에서 운영체제와 동작하고 있는 응용 프로그램의 보호를 위해 빠른 결함/복구 기능을 제공하는 것이다. 이 기능은 각각의 원격 메모리를 일정 크기의 메모리 블록으로 나누고, 그렇게 나누어진 메모리 블록들을 RAID5 기법을 응용하여

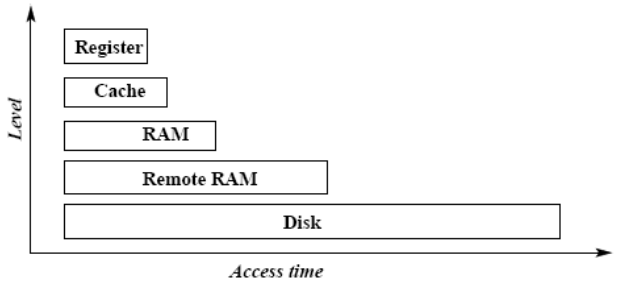

그림 1 새로운 메모리 계층에서 계층에 따른 다양한 접근 시간

결함내성 성질을 부여하여 원격 메모리를 구축할 수 있게 되는 것이다.

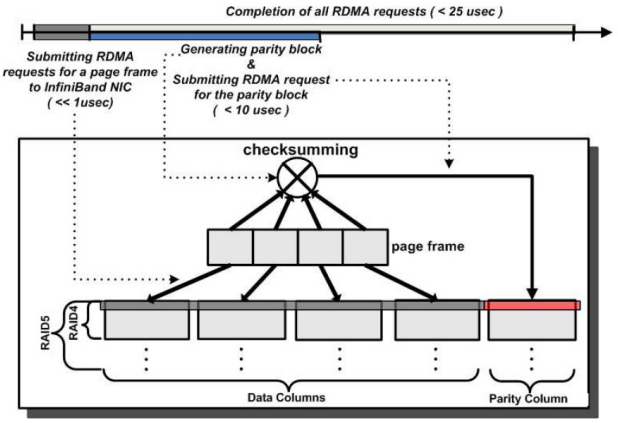

그림 2 원격 메모리 시스템에서 페이지 전송 원리

그림 2에서는 구축된 결함내성 메모리 시스템에서 원격 메모리에 페이지 전송을 하는 방식을 설명하고 있다. 하나의 메모리 페이지를 전송하기 위하여 먼저 원격 메모리 시스템은 메모리 페이지를 4개로 분할하여 4개의 1K 블록 데이터와 하나의 패리티(parity) 데이터를 5개의 각기 다른 원격 컴퓨터의 메모리에 전송을 수행하게 된다. 이 모든 과정들이 로컬 컴퓨터에서 원격 컴퓨터의 메모리를 직접 접근하여 수행하게 되므로 빠른 성능을 보장할 수 있는 것이다.

3. 실험 결과

실험을 위하여 대용량 메인 메모리 데이터베이스를 이용해서 TPC-C 벤치마크를 수행하였다. 실험은 총 8대의 컴퓨터를 이용하여 진행하였고, 6대의 원격 메모리 제공 컴퓨터와 1대의 서버 컴퓨터들은 수정된 linux-2.6.11 커널을 구동하도록 하였다. 또한 각 컴퓨터들은 RDMA 기능이 가능한 10 Gbps InfiniBand 카드를 장착하여 원격 메모리에 전송을 수행하게 된다. 그림 3에서와 같이 6대의 메모리 제공 노드들은 2.5GB 의 메모리를 서버노드들에 제공하고 있다. 그림 3는 실험을 위한 시스템 구성을 보여준다. front1 노드와 front2 노드에는 MySQL Cluster 소프트웨어 패키지를 설치하였으며, front1에는 mysqld, ndb_mgmd 프로세스가 front2는 ndbd 프로세스가 동작한다. ndbd가 실제 데이터를 메모리에 관리하는 프로세스이므로 front2에 수정된 linux-2.6.11이 수행된다. 반면, front1은 질의 처리 및 데이터베이스 시스템 관리를 위한 노드이므로 원래의 linux-2.6.11이 수행된다.

그림 4는 TPC-C 벤치마크를 수행한 결과이다. 비교를 위해서 front2에 메모리가 충분한 상황에서도 벤치마크를 수행했다(50MNS). 그림 5의 MNS는 메모리를 공유하지

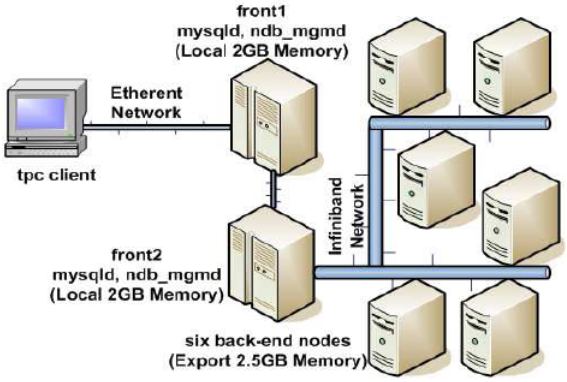

그림 3 원격 메모리 시스템 실험 구성

않은 상황을 표시하며, MS는 메모리를 공유하는 상황을 표시한다. WH는 TPC-C의 확장성을 표시하는 것으로서 WH의 값이 커질수록 요구하는 메모리가 커지게 된다. 그림에서 알 수 있듯이 메모리를 공유하더라도 우리 시스템의 성능이 크게 떨어지지 않는다.

# of WH	50 (MNS)	50 (MS)	75 (MS)	100 (MS)
Measured tpmC	3163	2940	2726	2610
Amount of Remote Memory		5.2GB	7.2GB	10.6GB
Ratio		92%	87%	82%

그림 4 TPC-C 벤치마크 결과

그림 5는 우리 시스템의 결함 내성 성질을 보여준다. 실험을 위하여 TPC-C 벤치마크를 수행하기 전에 한 번 그리고 75초 수행 중에 원격 메모리 제공 컴퓨터의 전원을 끊었다. 그림에서 알 수 있듯이, 시작 후 5초 동안 그리고 시작 후 80초에서 90초 사이에 TPC-C 벤치마크 프로세스가 전혀 동작하지 않다가 완전히 복구한 후에는 정상적으로 동작함을 확인할 수 있다.

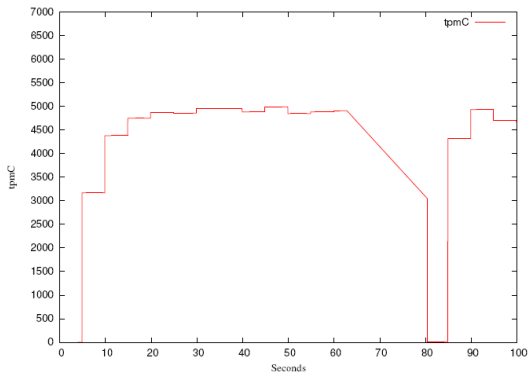

그림 5 결함 복구

4. 결론

이 논문은 원격 메모리 서버와 클라이언트로 이루어진 시스템에 결함 내성 성질을 부여하기 위한 소프트웨어 설계 및 구현을 설명하였다. 결함 내성을 위해서 메모리 한 페이지를 체크섬하는 방식을 사용하였다. 실험은 결함 내성 원격 메모리 시스템의 성능과 결함 내성 성질이 매우 우수함을 보여준다.

참고문헌

[1] D. Black, A. Gupta, and W-D Weber, “Competitive Management of Distributed Shared Memory", In Sprint COMPCON 89 Digest of Papers, Feburary 1989.

[2] W. Bolosky, M. Scott, and R. Fitzgerald, \Simple but Effective Techniques for NUMA Memory Management", In Proceedings of ACM Symposium on Operating System Principle, December 1989.

[3] M. Holliday, "Reference History, Page Size, and Migration Daemons in Local/Remote Architectures", In Proceedings of the International Symposium on Architectural Support for Programming Languages and Operating Systems, pages 104-112, April 1989.

[4] W. Bolosky, M. Scott, and R. Fitzgerald, R. Fowler, and A. Cox, "NUMA Policies and their Relationship to Memory Architecture", In Proceedings of the International Symposium on Architectural Support for Programming Languages and Operating Systems, pages 212-221, April 1991.

[5] Michael J. Freeley, William E. Morgan, Frederic H. Pighin, Anna R. Karlin, Henry M. Levy, "Implementing Global Memory Management in a Workstation Cluster", In Proceedings of ACM Symposium on Operating Systems Principles, December 1995.

[6] D. Comer and J. Griffoen, "A new design for distributed systems: The remote memory model", In Proceedings of the Summer 1990 USENIX Conference, June 1990.

[7] M. J. Frankling, M. J. Carey, and M. Livny,"Global memory management in client-server DBMS architectures", In proceedings of the 18th VLDB Conference, August 1992.

[8] M. D. Dahlin, R. Y. Wang, T. E. Anderson, and D. A. Paterson, "Cooperative caching: Using remote client memory to improve ¯le system performance", In Proceedings of the USENIX Conference on Operating Systems Design and Implementation, November 1994.