리눅스 클러스터 파일 시스템을 위한 Myrinet 기반 초고속 통신모듈의 설계 및 구현*

박의수[0]* 이홍기* 최현호** 김형식* 유원경*** 유관종*

*충남대학교 컴퓨터과학과, **대전보건대학 컴퓨터정보처리과, ***성신여대 컴퓨터정보학부

{uspark, helius, hskim, kjyoo}@cs.cnu.ac.kr, hyuno@hit.ac.kr, wyoo@cs.sungshin.ac.kr

Design and Implementation of a Myrinet based High-speed Communication Module for the Linux Cluster File System

Ui-Su Park[0]* Heung-Ki Lee* Hyun-Ho Choi** Hyong-Shik Kim* Weon-Kyung Yoo*** Kwan-Jong Yoo*

*Dept. of Computer Science, ChungNam University

**Dept. of Computer Information Processing, Daejeon health Science college

***Dept. of Computer Science, Sungshin Women's University

요 약

클러스터 파일 시스템은 데이터 입출력 대역폭을 극대화하여 효율성을 높이고 각 노드의 입출력 부담을 균등하게 부과하기 위하여 원본 파일을 여러 노드에 분산 저장한다. 이렇게 파일을 노드들에 분산 저장하기 위해서는 효율적인 노드간 데이터 통신을 필요로 하며, 노드 내부에서도 클러스터 파일 시스템과 어플리케이션과의 효율적인 전용 데이터 교환 메커니즘을 지원해야 한다. 본 논문에서는 WAN(Wide Area Network)에 적합하도록 설계된 TCP를 이용한 기존의 교환 메커니즘인 통신모듈이 가지고 있는 문제점을 해결하기 위해 다양한 프로토콜과 하드웨어적인 접근을 통해 Myrinet이 초고속 통신모듈에 적합함을 보이고 GM API를 활용하여 기존의 소켓기반인 TCP/IP를 이용한 통신 모듈을 대체할 새로운 통신모듈의 모델을 제시한다.

1. 서 론

클러스터 파일 시스템은 기존의 클러스터링 기술을 파일 시스템에 적용하여, 각 노드 단위로 파일 시스템을 구성할 때 발생되는 저장 공간과 대역폭의 제약문제를 극복하기 위한 방법이다[1]. 클러스터 파일 시스템이 안정적이고 효율적인 방법으로 대용량의 멀티미디어 데이터를 빠르게 분산 저장하기 위하여 필요한 교환 메커니즘인 통신 모듈은 클러스터 파일 시스템을 구성하는 각 노드의 데몬이 다른 노드에서 실행되고 있는 데몬과 자료를 주고받을 수 있도록 하여야 한다[2].

본 논문에서는 기존의 소켓기반의 TCP/IP를 이용한 통신 모듈이 가지고 있는 사용자 영역이나 커널 영역에 존재하는 여러 네트워크 계층으로 인하여 하부 네트워크가 제공하는 대역폭을 충분히 활용하지 못하여 생기는 호스트 내부의 소프트웨어구간의 병목현상으로 생기는 성능저하를 해결하고 클러스터 파일 시스템이 안정적이고 효율적인 방법으로 대용량의 멀티미디어 데이터를 보다 빠르게 분산 저장하기 위하여 필요한 통신 모듈을 설계한다. 이를 위해 기존의 프로토콜과 함께 Zero copy를 지원하는 사용자수준의 다양한 통신 방식의 성능을 다각적으로 분석하여 클러스터 파일 시스템상에서 멀티미디어 통신에 적합한 최적의 네트워크 통신환경을 결정한다. 다음으로 리눅스 클러스터의 핵심이 되는 클러스터 파일 시스템을 구성하는 각 노드간의 원활한 통신을 보장하기 위해 최적의 프로토콜과 하드웨어 조건에서 통신모듈의 구조를 정의하고 모듈간의 데이터 교환 과정을 설계한다. 마지막으로, 노드와 노드사이의 초고속 통신을 통한 데이터 교환을 가능하게 하는 노드간 통신모듈(Inter-node communication module), 즉 통신모듈 라이브러리를 최적의 프로토콜인 GM API[3]를 이용해 리눅스 환경에서 개발한다.

본 논문의 구성은 다음과 같다. 2장에서는 클러스터 파일 시스템에 적합한 최적의 프로토콜과 하드웨어적인 조건을 알아보기 위해 여러 통신 방식을 비교분석하며, 3장에서는 선정된 최적의 통신 프로토콜을 이용해 통신모듈을 설계 및 구현한다. 마지막으로 4장에서는 본 논문의 결론을 맺고 향후연구 과제를 기술한다.

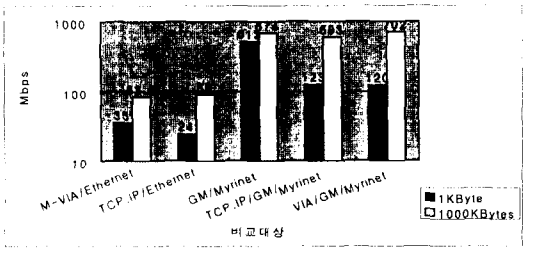
* 본 논문은 정보통신부 지원 선도기반기술개발사업에 의하여 수행된 과제의 결과임

2. 통신 방식 비교 분석

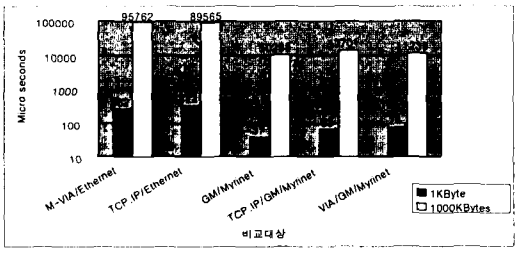
2.1 실험 구성

다양한 프로토콜과 하드웨어적인 실험을 위해서 아래와 같은 5가지 환경에서 실험을 시행하였다. 이더넷 기반에 각각 M-VIA와 TCP/IP를 대비시킨 실험을 통하여 M-VIA와 TCP/IP의 성능을 비교한다. 미리넷 기반에서는 각각 순수 GM, GM위에 TCP/IP, GM위에 VIA를 대비시켜 실험하여 이들간의 대역폭과 지연시간 비교 분석한다. 이를 통하여 다양한 통신 환경 중에서 클러스터 파일 시스템에 적합한 통신 환경을 채택한다.

- M-VIA over Ethernet
- TCP/IP over Ethernet
- GM over Myrinet
- TCP/IP on GM over Myrinet
- VIA on GM over Myrinet



[그림 1] 비교대상별 대역폭



[그림 2] 비교대상별 지연시간

2.2 실험 결과

[그림 1]과 [그림 2]는 단위 패킷 1K바이트와 1M바이트에 해당하는 대역폭과 지연시간에 대한 펜티엄IV 1.71Ghz와 펜티엄III 866Mhz간의 비교대상별 실험결과이다. 미리넷 기반의 GM이 1KByte의 작은 사이즈부터 1MByte의 큰 사이즈에 대해서 가장 좋은 성능을 보인다. 그러나 미리넷 기반의 GM위의 VIA는 100K바이트 미만의 작은 사이즈에 대한 대역폭이 미리넷 기반의 GM에 비해 현저히 떨어지므로 미리넷 기반의 GM이 새로운 통신모듈의 설계에 가장 적합함을 알 수 있다.

3. GM 기반 초고속 통신 모듈의 설계 및 구현

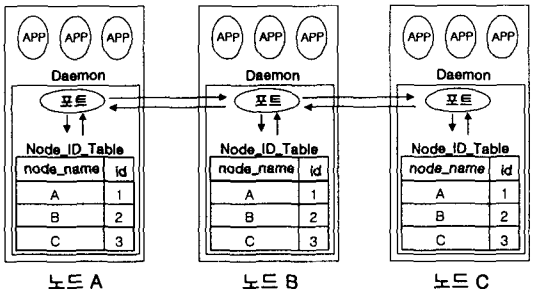
3.1 GM 통신 라이브러리의 설계 요구사항

GM 통신 특성을 데몬 개발자 수준 GM API에 적용하여야하며 개발하고자 할 GM 통신모듈의 통신특징은 아래와 같다[4].

- GM에서의 데몬간의 통신은 비연결형(Connectionless)이기 때문에 상대 데몬을 아이디(ID)를 통하여 인지하여 메시지를 처리하며 흐름제어 한다. 즉, 송신노드는 메시지를 전송하기 위해, 포트를 열고 수신준비를 마친 데몬에게 자신의 아이디와 함께 데이터를 전송하며, 수신데몬은 상대 아이디를 구분하여 이를 처리하여 메시지를 수신한다. 이때 GM은 GM 드라이버 초기 설정 시 매퍼(Mapper)라는 응용프로그램을 이용해 스위치를 통해 연결된 노드들에게 노드의 고유 아이디 값을 부여하고 부여된 아이디 확인을 통하여 통신을 한다. 이와 같은 특성을 이용하여 설계하기 위해서는 호스트 아이디와 GM 아이디를 구분하여 인지할 수 있도록 하여야 한다.
- GM은 두 가지의 메시지 우선순위 레벨(Level)을 제공하여 GM 통신모듈은 높은 우선순위를 가진 메시지의 인터럽트를 통하여 데드락(Deadlock)을 효과적으로 방지할 수 있다.
- TCP/IP방식에서는 리스너스레드(Listener Thread)를 통하여 응용프로그램이나 다른 노드의 서비스 요청을 처리와 동시에 다른 노드의 TCP 연결 요청에 대한 수락이 가능 하다. GM API에서는 이를 대처할 방안을 제시하여야 한다.

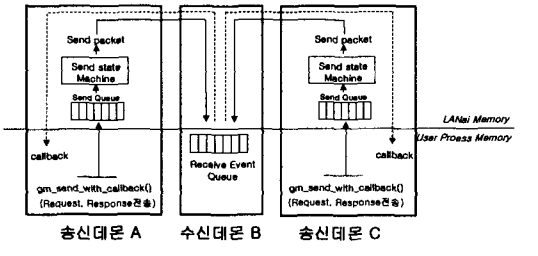
3.2 GM통신모듈 구조 설계

[그림 3]은 GM통신모듈의 구조를 나타낸다. GM 통신 라이브러리는 그림과 같이 데몬 간의 통신수행 부분을 정의한다. 클러스터 파일 시스템의 구성 노드는 GM통신모듈을 이용하여 다른 노드와 통신을 수행한다. 노드 간 통신은 기본적으로 GM API를 사용하며 클러스터 파일 시스템 데몬을 실행하면 데몬 초기화 단계에서 다른 노드의 데몬과 GM 연결을 준비한다. 이미 실행 중인 데몬은 새로운 데몬의 GM 연결 요청을 받고, 연결 준비된 데몬과 필요 시 데이터 교환을 한다. 데몬은 별도의 리스너 스레드를 통한 응용프로그램이나 다른 노드의 서비스 요청을 처리할 필요 없이 동시에 다른 노드의 GM 연결 요청을 수락 할 수 있다.



[그림 3] GM 통신모듈을 이용한 통신 구조 설계

3.2.1 메시지 송신 내부 흐름

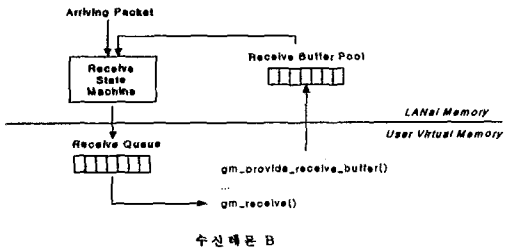


[그림 4] 메시지 송신 내부 흐름

[그림 4]는 송신데몬에서 수신데몬에게 메시지를 전송하는 내부 흐름도 이다. 송신데몬 A와 송신데몬 C는 각각 수신측 노드 아이디와 메시지를 콜백(callback) 요청과 함께 gm_send_with_callback()를 통하여 메시지를 전송하면 이 메시지는 LANai 메모리의 큐(Queue)를 통하여 Send state Machine, 즉 MCP(Myrinet Control Program)에서 처리되어 수신데몬 B에 전송된다. 이때 수신데몬 B는 콜백 응답을 각 데몬에게 보내주게 되며 이를 통하여 비연결형 통신방식인 GM이 신뢰성을 지닐 수 있다[4].

3.2.2 메시지 수신 내부흐름

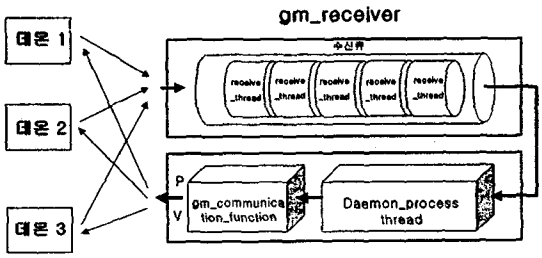
[그림 5]는 송신데몬에서 보낸 메시지를 수신하는 내부 흐름도이다. 수신데몬은 사용자 가상 메모리(User Virtual Memory) 즉, 응용프로그램에서 소프트웨어의 작동에 의해 접근할 수 있는 메모리상에서 gm_provide_receive_buffer()를 통해 LANai 메모리상의 수신 버퍼풀을 Receive State Machine에 제공하면 Receive State Machine은 수신된 메시지를 처리하여 사용자 가상메모리상의 수신 큐를 거쳐 gm_receive()를 통해 폴링(Polling)방식으로 메시지를 처리한다.



[그림 5] 메시지 수신 내부 흐름

3.2.3 GM통신모듈의 다중노드 지원

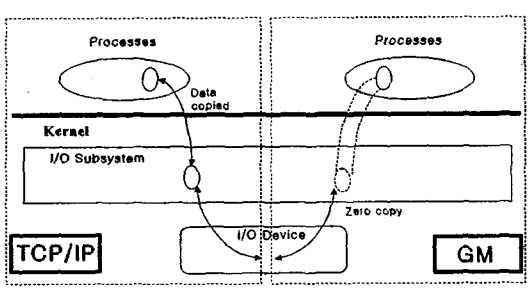
GM을 통한 다중노드간의 통신을 위해서는 서로 다른 여러 개의 데몬으로부터 오는 요청이나 응답을 구분하여 처리 할 수 있어야 한다. 이를 위해서 각 데몬으로부터 전해져오는 요청이나 응답에 대해 receive_thread를 생성하고 큐를 통하여 순서대로 수신하며, 이를 Daemon_process_thread에서 각각의 메시지를 구분하여 해당 노드의 데몬에게 송신한다. 이때 세마포어(Semaphore)를 이용하여 전송이 완료된 후에 다음 데이터를 전송 할 수 있도록 한다.



[그림 6] GM 통신모듈의 다중노드 지원

3.2.4 GM을 이용한 Zero copy의 응용

[그림 7]은 TCP/IP를 이용한 통신모듈에서 프로세스와 I/O 디바이스간 커널의 개입으로 인한 오버헤드 발생을 GM통신모듈의 메시지 전송시 내부적인 메시지 복사과정에서 Zero copy를 통하여 오버헤드 발생을 줄이는 과정을 보여 준다. 응용프로그램의 요청을 데몬 자신이 응답을 해줄 수 없는 경우 다른 노드의 데몬에게 요청을 보내는 _forward_request()함수를 통한 노드간의 통신에서 기존의 TCP/IP방식 통신모듈에서는 프로세스와 I/O 디바이스사이에 커널의 I/O 서브시스템(I/O Subsystem)에서 한번의 데이터복사가 이루어진다. 하지만 GM 통신모듈에서는 프로세스와 I/O 디바이스간 Zero copy가 수행되어 복사에 의한 통신 오버헤드를 줄인다.



[그림 7] GM 통신상에서의 Zero copy

4. 구 현

본 통신 모듈이 구현된 환경은 [표 1]과 같다.

[표 1] 구현환경

운영체제	리눅스 커널 2.2.x 혹은 2.4.x
컴파일러	g++, Make 유틸리티
네트워크	GM over Myrinet

통신모듈이 제공하는 API 함수들의 리스트는 [표 2]와 같다.

[표 2] 통신모듈이 제공하는 함수들

구분	함수이름	내용
응용프로그램과 데몬간의 통신을 위한 함수	_send_request	요청전달 함수
	_receive_response	데몬으로부터 응답을 받는 함수
노드와 노드간의 통신을 위한 함수	_receive_message	응용프로그램이 다른 노드의 데몬으로부터 요청 혹은 응답을 받는 함수
	_forward_request	요청을 다른 노드의 데몬에게 보내는 함수
	_send_response	응답을 동일 노드의 응용프로그램에게 보내는 함수
	_forward_response	응답을 다른 노드의 데몬에게 보내는 함수

5. 결론

본 논문에서는 클러스터 파일 시스템에서 흔히 사용되는 기존 소켓기반의 TCP/IP를 이용한 통신모듈에서의 단점인 대용량의 데이터에 대한 노드간의 전송속도 문제점을 해결하기 위하여 실험을 통하여 클러스터 파일 시스템에 적합한 최적의 사용자 수준의 통신 프로토콜을 선정하였다. 그리고 선정된 통신 프로토콜과 하드웨어적 환경인 Myrinet GM의 타당성을 제시하고 이 GM API를 이용하여 통신모듈의 기능을 효과적으로 구현하는 방안을 제시한 후에 초고속 통신모듈의 모델을 제시하고 구현하였다.

본 논문에서 제시한 GM기반 통신모듈은 프로세스와 I/O 디바이스간 커널의 개입을 제거한 Zero copy에 입각하여 복사에 의한 통신 오버헤드를 줄일 수 있었다. 본 연구 결과를 토대로 좀더 다양한 네트워크 환경 하에서의 통신 성능 측정이 필요하다. 그리고 통신 모듈의 최적화의 요건이 되는 다양한 패킷사이즈에 대한 테스트와 함께 최고의 성능을 발휘할 수 있는 버퍼 관리기법을 통하여 성능을 개선할 계획이다.

참고문헌

[1] Rajkumar Buyya, High Performance Cluster Computing Programming and applications, Volume 2, 1999

[2] Stephen E. Deering and David R. Cheriton, "Multicast Routing in Datagram Internetworks and Extended LANs", ACM Transactions on Computer Systems, Vol. 8, No. 2, May. 1990

[3] Dimitrov, Rossen. A Windows NT Kernel-Mode Device Driver for PCI Myrinet LANai 4.x Interface Adapters. M.S. Project Report, Mississippi State University, 1997, http://www.cs.msstate.edu/publications/theses_and_dissertations.html

[4] Myricom, GM Documentation, Available from http://www.myri.com/GM/doc/gm_toc.html (10 February 1998).