그리드 환경을 위한 데이터 분산 전송 기법

A Data Dissemination Mechanism for Grid Environments

김 형 진* 이 종 숙**
Hyungjinn Kim Jongsuk Ruth Lee

요 약

다양한 그리드 커뮤니티가 생겨나면서 그리드 환경에서의 데이터 분산 전송의 요구가 증가하고 있다. 기존 LAN 환경 또는 특수 목적 네트워크망에서는 데이터 분산 전송을 위해서 멀티캐스팅 기술을 활용하여 네트워크 트래픽을 많이 줄일 수 있었다. 하지만 전송 성능이 일반 유니캐스트 전송보다 좋지 못하고, 하드웨어 지원 없이는 일반적인 WAN 환경에서 멀티캐스팅을 적용할 수 없다는 단점을 갖고 있어서 그리드 응용에 적용하는 데는 한계가 있다. 이러한 문제점을 극복하기 위해서 본 논문에서는 데이터 분산 전송 기법인 DDMG(Data Dissemination Mechanism for Grid)를 제안하였다. DDMG는 보다 신속한 데이터 전송을 위하여 P2P 데이터 공유 방법을 활용하였으며, 또한 Globus XIO 라이브러리를 활용하여 그리드 환경의 다양한 프로토콜을 지원하도록 설계하였다. 본 논문에서는 DDMG 기법을 구현하여 LAN 환경과 WAN 환경에서 일반 유니캐스트 전송 방법과 비교 분석하였다.

Abstract

Nowadays as many Grid communities appear, the requirement of singlepoint-to-multipoint data transfer in Grid environments is growing. In a typical LAN or a special purpose network environment, a multicasting technology was used in such a data dissemination transfer case. However, compared with unicasting transfer performance the multicasting transfer is worse, and the obligation of a special hardware setting makes it difficult to implement in common Grid environments. Therefore, in this paper we propose an effective data dissemination mechanism for Grid environments named DDMG(Data Dissemination Mechanism for Grid). DDMG uses P2P(Peer-to-Peer) mechanism and Globus XIO library to improve the performance in a data dissemination process and to support heterogeneous protocols in Grid environments. We also evaluate the performance of DDMG mechanism in LAN and WAN environment by comparing with unicast transfer.

☞ Keyword : 그리드, 데이터 분산 전송, P2P(Peer-to-Peer), Globus XIO(Globus eXtended Input/Output)

1. 서 론

고에너지 물리학, 생명공학, 기상예보와 같은 주요 그리드 응용분야는 데이터 집약적 연구라는 특성을 가지고 있다[1-4]. 즉 고에너지 물리 연구에는 연간 수 백 페타바이트의 데이터가 생성되고, 기상예보에서는 보통 수 백 기가바이트의 데이터가 생성된다. 이에 대용량의 데이터를 전송 및 저장할 수 있는 DPSS(Distribued Performance Storage System), HPSS(High Performance Storage System)와 DFS(Distributed File System) 같은 다양한 기술들이 생겨났으며, 이러한 기술들은 각각 독자적인 저장장치 및 프로토콜을 갖게 되었다. 이렇게 저장기술이 다양해짐에 따라 응용 과학자들은 지역적으로 분산된 데이터 저장장치와 컴퓨팅 자원을 묶어주는 데이터 그리드 기술을 연구 개발하게 되었다[5].

데이터 그리드란 지역적으로 분산된 이기종의 저장 장치 자원들을 가상의 단일 자원으로

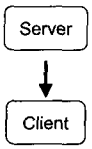
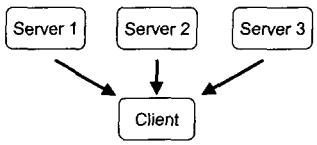
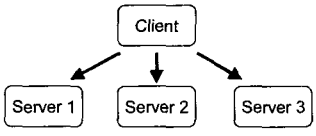
* 준회원: 한국과학기술연합대학원대학교
qanii@kisti.re.kr
** 정회원: 한국과학기술연합대학원대학교 겸임교수
한국과학기술정보연구원 슈퍼컴퓨팅센터
그리드컴퓨팅연구팀
jsruthlee@kisti.re.kr
[2006/08/25 투고 - 2006/09/06 심사 - 2006/10/09 심사완료]

통합함으로써 사용자에게 공통된 표준 인터페이스를 제공하는 기술을 말한다. 이는 특히 WAN(Wide Area Network) 환경에서 이러한 인터페이스를 효율적이고 안정적인 방법으로 제공하는 것을 목적으로 하고 있다. 이를 위해 데이터 그리드에서 가장 중요하게 다루어지는 기술 중의 하나가 데이터 전송 기술이다. 그리드 환경에서의 데이터 전송은 대용량의 데이터를 다양한 프로토콜을 이용하여 원거리의 자원들 간에 보다 효율적이며 안정적으로 이루어져야 한다. 하지만, 기존의 HTTP(Hyper-Text Transfer Protocol)나 FTP(File Transfer Protocol)는 WAN 환경에서의 이러한 요구사항을 만족시키는데 한계가 있어, 그리드 연구자들은 그리드 환경에 보다 적합한 새로운 프로토콜 개발에 힘써왔다[6].

ANL(Argone National Laboratory)에서는 위와 같은 요구사항들을 충족하기 위해 GridFTP (Grid File Transfer Protocol)를 개발하였다[7]. GridFTP는 그리드 환경에서 대용량 데이터를 안정적이며 고속으로 전송할 수 있도록 기존의 표준 FTP를 확장한 프로토콜이다. 이 프로토콜은 병렬 데이터 전송(Parallel Data Transfer), TCP 버퍼/윈도우 크기 자동조절 등 그리드 환경에 적합한 여러 가지 데이터 전송 기능들을 제공하고 있으며, 이 외에도 Kerberos[8]와 같은 보편적으로 널리 사용되고 있는 보안 기술뿐만 아니라 최신 그리드 보안 기술인 GSI (Grid Security Infrastructure)[9]도 지원한다. 이러한 여러 가지 이점 때문에 EU의 EGEE (Enabling Grid for E-sciencE) 프로젝트[10]나 미국의 TeraGrid 프로젝트[11] 같은 대표적인 국제 그리드 프로젝트에서는 GridFTP를 주 전송 프로토콜로 사용하고 있다.

위에서 언급한 바와 같이 GridFTP는 다양한 기능들을 제공하고 있지만, 지원하지 못하는 기능들도 있다. GridFTP의 주요 기능들을 데이터 전송 형태별로 분류하면 <표 1>과 같다. GridFTP는 SSSC(Single Server to Single Client)와 MSSC(Multi Server to Single Client) 형태의 전송에는 다양한 기능들을 제공하고 있으나, SCMS(Single Client to Multi Server) 형태의 전송에는 어떠한 전송 기능도 제공하고 있지 않다. 문제는 이러한 SCMS 형태의 데이터 분산 전송이 그리드 응용분야에서 활용도가 점차 증가하고 있다는 점이다. 일례로, 한 장소에서 생성된 수 백 테라바이트의 데이터를 지역적으로 분산된 여러 저장장치에 전송해야 하는 고에너지 물리 연구가 있으며, 또한 대형 시뮬레이션 장치로 생성된 수 백 테라바이트의

〈표 1〉 데이터 전송 형태별 GridFTP의 지원 기능

데이터 전송 형 태	SSSC (Single Server to Single Client)	MSSC (Multi Server to Single Client)	SCMS (Single Client to Multi Server)
데이터 전송 구 조			
GridFTP의 기 능	• 병렬 데이터 전송 • TCP 버퍼/윈도우 크기 자동 조절 • 제 3자 파일 전송 제어 • 파일 부분 전송	• 스트라이프 모드 데이터 전송	• 해당 기능 없음

데이터를 지역적으로 분산된 여러 컴퓨팅 사이트로 전송해야 하는 TeraGyroid 프로젝트[12] 등이 있다. 현재까지 이러한 프로젝트에서는 데이터를 전송하기 위하여 초고속 네트워크망을 바탕으로 기존의 GridFTP 프로토콜을 이용하였으나, 프로젝트의 규모가 점차 커짐에 따라 보다 효율적이며 안정적인 데이터 분산 전송 기법의 필요성이 대두되고 있다. 이에 본 논문에서는 그리드 환경에서 보다 효율적이고 신속하게 데이터를 분산 전송할 수 있는 기법을 제안하였다.

본 논문의 구성은 다음과 같다. 제 2장에서는 기존의 관련 연구로써 데이터 분산 전송 기술과 그리드 환경에서 다양한 프로토콜의 지원을 가능하게 해주는 Globus XIO[13] 라이브러리에 대해서 알아본다. 제 3장에서는 본 논문에서 제안한 데이터 전송 기법인 DDMG(Data Dissemination Mechanism for Grid)의 구조와 특징에 대해서 알아보고, 제 4장에서는 DDMG 구현물의 성능을 일반 유니캐스트 전송 기법과 비교 및 분석하였다. 그리고 제 5장에서 결론을 맺는다. 추가적으로 부록 A 에서는 DDMG 서버/클라이언트의 유사부호(Pseudo Code)에 대해서 기술 하고, 부록 B에서는 DDMG의 단점인 디스크 입출력 부하에 대해서 기술한다.

2. 관련 연구

본 장에서는 그리드 환경에서의 멀티캐스트 전송 기술과 그 기술들의 장단점에 대해서 기술하고, 최근 몇 년 사이에 데이터 공유를 목적으로 사용 빈도가 급격히 증가한 P2P(Peer-to-Peer) 전송 기술을 소개한다. 끝으로 이기종 자원으로 구성된 그리드 환경에서 다양한 프로토콜을 하나의 통일된 인터페이스로 통합해 주는 Globus XIO 라이브러리에 대해서 기술한다.

2.1 데이터의 분산 전송 기술

2.1.1 멀티캐스트 전송

LAN(Local Area Network) 환경에서는 데이터를 효율적으로 분산 전송하기 위해서 멀티캐스트 전송 기술을 주로 사용하고 있다. 이 기술은 데이터의 분산 전송 시 유니캐스트 전송에 비하여 네트워크 부하를 크게 줄일 수 있다는 장점을 갖고 있어 세계 곳곳에서 많은 연구가 활발히 이루어지고 있으며, 또한 다양한 프로토콜이 개발 되고 있다. 하지만 인터넷 상에 있는 대부분의 라우터들은 멀티캐스트 전송의 남용을 막기 위하여 멀티캐스트 기능을 지원하지 않으며, 또한 그리드 환경을 고려하여 개발된 멀티캐스트 전송 프로토콜은 있으나, 실제 응용에 적용하여 활용하기에는 문제점들이 있다.

그리드 환경을 위하여 개발되어진 멀티캐스트 전송 프로토콜로는 TCP-XM(TCP eXtended to support Multicast)[14]이 있다. TCP-XM은 데이터 분산 전송을 위해 개발된 프로토콜이며 데이터 전송 시 멀티캐스트 환경을 감지하여 자동적으로 멀티캐스트 전송으로 전환되도록 해주는 프로토콜이다. 이 프로토콜의 특징으로는 전송의 신뢰성을 확보하기 위하여 멀티캐스트 전송 프로토콜을 TCP 기반으로 구현하였다는 점과 사용자 계층 라이브러리인 lwIP(light-weight TCP/IP stack) 프로토콜[15] 기반으로 구현하여 커널 컴파일 등의 직접적인 운영체제 코어 코드의 수정 없이 용이하게 설치 및 활용할 수 있도록 개발되었다는 점이다. 이렇게 개발된 TCP-XM은 데이터를 분산 전송함에 있어서 일반 유니캐스트 전송에 비하여 네트워크 트래픽 부하를 상당히 줄일 수 있다는 장점이 있으나, 전송 속도 측면에서는 일반적인 TCP 전송보다는 좋지 못하다는 단점을 가지고 있다.

TCP-XM 외에 그리드 환경을 고려하여 개발되어지지는 않았지만 그리드 환경에서의 활용

가능성이 높은 프로토콜로는 MDP(Multicast Dissemination Protocol)[16], NORM(Nack-Oriented Reliable Multicast)[17], LGMP(Local Group-based Multicast Protocol)[18] 등이 있다. MDP 프로토콜은 RM(Reliable Multicasting)을 목적으로 개발된 프레임워크 프로토콜이며, 멀티캐스트 데이터 전송에서 신뢰성을 보장하기 위하여 NACK(Negative Acknowledgements) 기술을 이용하고 있다. NORM 프로토콜은 이기종의 IP 네트워크와 프로토콜에서 효과적인 데이터 전송을 가능하게 하기 위해여 개발되어진 프로토콜이다. 이 프로토콜 또한 멀티캐스트 데이터 전송의 신뢰성을 보장하기 위하여 MDP와 비슷한 NACK 기술을 사용하고 있으며, 현재 IETF의 멀티캐스트 프로토콜 워킹그룹에 의해 표준화되었다. LGMP는 로컬 그룹(Local Group) 이론을 적용한 RM 프로토콜이다. 즉, LGMP는 로컬 그룹이라는 서브 그룹을 동적으로 형성해서 이들 중 그룹 컨트롤러(Group Controller)를 지정하여 전송 실패와 각종 피드백 메시지의 전송을 관리한다는 특징을 가지고 있다. 이들 프로토콜들의 공통된 특징으로는 오픈소스로 개발되어 있어 여러 다양한 플랫폼을 지원하고, 라우터의 설정 없이 수신자 지정 및 확장이 가능하며, 코드의 완성도가 상당히 높다는 장점을 가지고 있다. 하지만 이러한 프로토콜도 유니캐스트 전송에 비하여 전송 속도 측면에서 성능이 좋지 못하며[19], WAN 환경에서 이 기술들을 활용하기 위해서는 MBone(Mulicasting Back-bone)[20]과 같은 기술을 사용하여 가상 멀티캐스팅 환경을 구축해야 한다는 문제점이 있다.

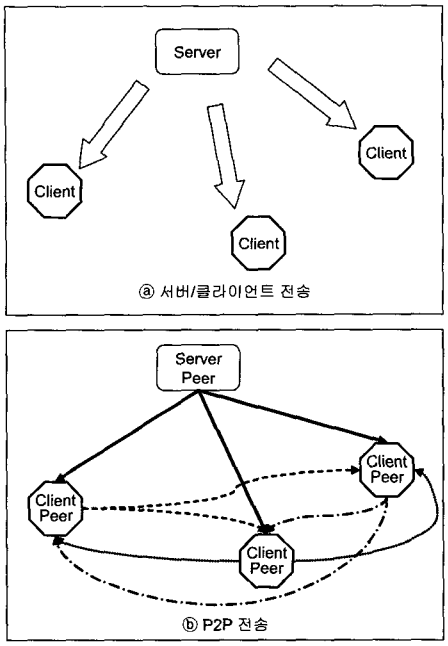
2.1.2 P2P 데이터 공유 기술

유니캐스트 전송 기술을 약간 변형하여 인터넷 상에서 보다 신속하게 데이터의 배포 및 공유를 가능하게 하는 P2P(Peer-to-Peer) 데이터 공유 기술이 최근에 널리 사용되고 있다[21, 22]. 기존의 서버/클라이언트 형태를 갖는 유니캐스트 전송 기술은 모든 전송 서비스가 서버에 집중되어 있어 클라이언트가 많아지면 네트워크 대역폭의 한계로 전송 속도가 현저하게 저하되는 단점을 갖고 있었다. 이러한 단점을 극복하기 위하여 멀티캐스트 전송기법이 개발되어, 네트워크 부하를 줄일 수 있었으나, 전송 성능이 좋지 못하며 범용적인 인터넷 환경에서는 사용할 수 없다는 단점을 갖고 있다.

P2P 데이터 공유 기술은 이러한 전형적인 서버/클라이언트의 형태를 벗어나 네트워크에 참가하는 모든 클라이언트가 동시에 서버의 역할도 수행한다는 데에 특징이 있다. 즉, 각각의 클라이언트들은 자기가 받은 데이터의 일부를 다른 클라이언트와 주고받으며 전체 데이터를 전송 및 공유하게 된다. 이렇게 데이터 전송 서비스를 데이터의 전송에 참가하는 모든 서버 및 클라이언트들에게 분산함으로써 네트워크 자원을 보다 효율적으로 활용할 수 있다는 장점을 가지고 있다. 또한, 클라이언트 입장에서는 데이터를 여러 서버에서 받는 효과를 얻을 수 있어 보다 신속한 전송이 가능하며, 멀티캐스트 전송과는 다르게 추가적인 하드웨어 설정 없이 일반 인터넷 환경에서도 범용적으로 사용할 수 있다는 장점을 가지고 있다. <그림 1>은 각각 일반 유니캐스트 서버/클라이언트 전송과 P2P 전송을 나타내고 있다.

2.2 Globus XIO

그리드 환경에서는 다양한 저장 장치와 프로토콜들 간에 정보를 효과적으로 주고받기 위해서 하나의 통일된 통신 인터페이스가 요구된다. 이러한 통일된 인터페이스를 제공하기 위해서 ANL에서는 Globus XIO(eXtensible Input /Output)[13]를 개발하였다.



〈그림 1〉 서버 클라이언트 전송과 P2P 전송

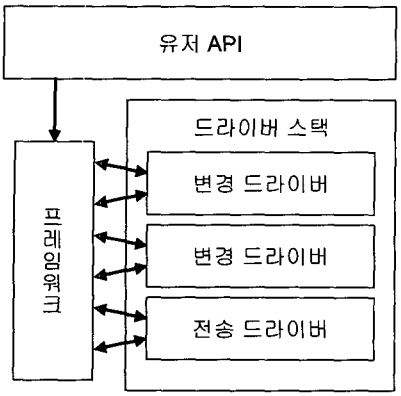
Globus XIO는 다양한 입출력 라이브러리를 하나의 통합된 인터페이스로 나타내기 위한 프레임워크 라이브러리이다. 이는 확장성있는 단순하고 직관적인 입출력 API(Application Program Interface)를 제공하고 있으며, 다음과 같은 2가지 주된 목적을 위해서 개발되었다.

- 그리드 환경의 모든 입출력 프로토콜에 대하여 하나의 일관된 사용자 API의 제공
- 새로운 프로토콜 개발에 소요되는 노력과 시간의 최소화

기존에는 다양한 통신 프로토콜을 지원하는 어플리케이션을 개발하기 위해서 서로 다른 다양한 프로토콜 API를 활용해야 했다. 이 때문에, 개발 코드가 복잡해지고 프로토콜의 추가 또는 확장이 어려웠다. 하지만 Globus XIO를 활용하면 다양한 프로토콜을 서로 같은 인터페이스로 쉽게 구현할 수 있어 복잡성을 최소화 할 수 있으며 프로토콜의 확장도 용이하다. 이러한 통일된 프레임워크는 새로운 프로토콜을 개발함에 있어 프레임워크 디자인에 소요는 시간을 단축하고 개발 시간을 프로토콜 핵심 기능 개발에 최대한 집중할 수 있도록 해준다.

<그림 2>는 Globus XIO의 구조를 나타낸다. 아래 그림에서 '*유저 API*'는 OCRW(open/close/read/write) 기능을 갖는 Globus XIO의 사용자 외부 인터페이스, '*프레임워크*'는 Globus XIO 자체 코어 라이브러리, '*드라이버 스택*'은 사용할 드라이버를 등록할 수 있는 공간, '*드라이버*'는 특정 프로토콜의 코어 기능을 수행하는 모듈이다. 드라이버는 '*전송 드라이버(transport driver)*'와 '*변경 드라이버(transform driver)*' 크게 두 가지로 분류되며, '*전송 드라이버*'는 데이터의 전송 기능을 하는 드라이버이며 '*변경 드라이버*'는 데이터의 가공 기능을 하는 드라이버이다. 일례로 TCP(Transmission Control Protocol) 및 UDP(User Diagram Protocol) 드라이버는 각각 TCP 전송 기능과 UDP 전송 기능을 수행하는 전송 드라이버이며, GSI 드라이버는 데이터의 암호화를 담당하는 변경 드라이버이다. <그림 2>와 같이 드라이버 스택에는 여러 개의 드라이버를 동시에 올려서 사용할 수가 있다. 하지만 전송 드라이버는 반드시 하나만 존재하여야 하고, 변경 드라이버는 한 개 이상 또는 없을 수도 있다.

이러한 구조는 Globus XIO로 구현된 어플리케이션에 새로운 프로토콜을 적용하려 할 때 코드 수정을 최소화 할 수 있다는 장점을 갖는다. 하지만 유저 API가 OCRW 구조로 고정되어있어 API의 확장성이 결여되었다는 단점도 갖고 있다. Globus Toolkit에서 기본적으로 제공되는 전송 드라이버로는 TCP, UDP, MODE E, HTTP, FILE 드라이버가 있으며, 변경 드라이버로는 GSI, ORDERING 드라이버가 있다. 특히 기본 전송 드라이버인 TCP, UDP 드라이버는 일반 소켓 통신과 성능 차이가 거의 없다는 연구 발표가 있어 Globus XIO의 성능을 입증하고 있다[23].



〈그림 2〉 Globus XIO의 구조

본 논문에서 제안한 데이터 분산 전송 기법은 보다 범용적인 WAN 환경에서 다양한 프로토콜의 지원을 위하여 P2P 데이터 전송 기술 및 Globus XIO 라이브러리를 활용하여 구현하였다.

3. DDMG의 설계 및 구현

본 논문에서 제안하는 전송 기법인 DDMG는 원시 데이터를 가지고 있는 하나의 클라이언트에서 동시에 여러 서버에게 데이터를 전송함에 있어 일반 유니캐스트 전송 기술보다 효율적이면서 신속한 전송을 하는 데에 목적이 있다. 이를 위하여 본 기법은 P2P 데이터 전송 기술을 활용하여 전송 속도 및 네트워크 대역폭 이용의 효율성을 개선하였다. DDMG는 전송의 주체인 하나의 클라이언트와 하나 이상의 서버로 구성되어진다. 클라이언트/서버간의 데이터 전송은 클라이언트에서 서버로만 이루어지며 이들 사이의 통신은 Globus XIO 라이브러리를 사용하여 다양한 프로토콜의 지원이 가능하도록 하였다.

본 장에서는 DDMG의 데이터 전송 과정에 대해서 소개하며, 그리고 DDMG를 활용하여 구현한 서버/클라이언트 툴에 대해서 소개한다.

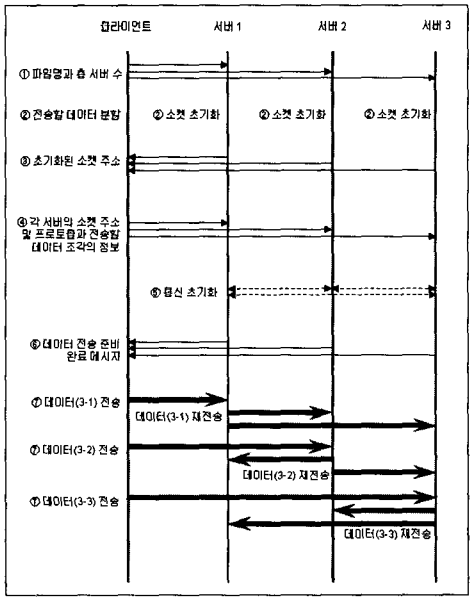
3.1 DDMG의 데이터 전송 메커니즘

일반 P2P 데이터 공유 기법은 불특정 다수의 피어(Peer)들이 불특정 시간동안 서로의 데이터 조각을 주고받으며 특정 데이터를 공유하는 데에 목적이 있다. 그러므로 각각의 피어들은 데이터 전송을 자체적으로 스케줄링해야 한다. 이로 인해 피어들 간에 데이터 중복 전송이 일어날 수도 있으며, 각 피어들의 데이터 상태정보를 얻기 위한 추가적인 네트워크 트래픽이 불가피하다. 즉 일반적인 P2P 데이터 공유 기법으로는 효율적인 데이터 분산 전송을 하는 데에 있어서 한계가 있다.

이러한 문제점을 극복하기 위해서 DDMG는 데이터 전송 주체인 클라이언트가 특정 다수의 서버에게만 데이터 전송이 이루어지도록 하였다. 이렇게 데이터를 전송하기 전에 수신 서버를 정함으로서, 데이터 분산 전송을 최적화 할 수 있으며, 기존의 P2P 데이터 공유 기법의 피어 검색과 같은 데이터 전송과 관련 없는 기능들을 배제할 수 있고, 서버간의 데이터 상태정보 메시지 등의 불필요한 네트워크 트래픽을 최소화 할 수가 있어서, 보다 효율적인 데이터 전송이 가능하게 된다.

<그림 3>은 DDMG의 데이터 전송 메커니즘을 나타낸다. 클라이언트는 사용자로부터 파일명, 전송할 서버주소와 프로토콜을 입력받은 후에, 입력받은 파일명과 총 서버의 수를 각각 서버들에게 전달한다. 각각 서버들은 전달 받은 정보를 바탕으로 이웃 서버(자신 이외의 다른 모든 서버)들로부터 데이터를 전송받을 통신 소켓을 초기화하며, 초기화된 소켓 주소는 다시 클라이언트에게 전달한다. 그동안 클라이언트는 전송할 파일을 전송할 서버 수만큼 나누며, 나눠진 각 파일 조각의 시작 주소와 끝 주소, 그리고 클라이언트로 전달 받은 이웃 서버의 소켓 주소를 각각의 서버에게 전달한다. 각각의 서버들은 이러한 정보를 바탕으로 전송

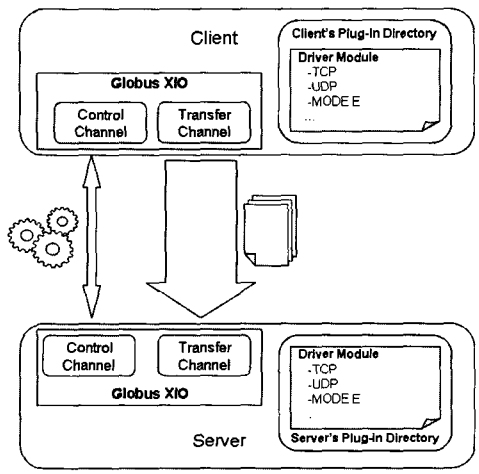
받을 파일을 초기화하며, 이웃 서버와 통신을 초기화 한다. 이러한 일련의 과정이 끝나면 클라이언트에게 "데이터 전송 준비 완료" 메시지를 전달한다. 이 메시지를 전달 받은 클라이언트는 각각의 서버에게 데이터 전송을 수행한다 (보다 상세한 유사부호(Pseudo code)는 부록 A의 <표 A-1>과 <표 A-2>를 참조).



〈그림 3〉 DDMG의 데이터 전송 메커니즘

3.2 DDMG의 구현

DDMG의 구현은 Linux 환경에서 C언어로 구현하였으며, Globus Toolkit 4.0.2의 Globus XIO 라이브러리를 활용하였다. 본 툴의 통신 구조는 컨트롤 채널과 데이터 전송 채널로 나눠진다. 컨트롤 채널은 데이터를 전송하는데 필요한 데이터의 정보와 클라이언트/서버 간의 각종 상태 정보를 주고받는데 쓰이는 소켓 스트림이며, 전송 채널은 실질적으로 데이터가 전송되는 소켓 스트림이다. 이 두 채널 모두 Globus XIO 프레임워크를 사용하고 있어 사용자 임의의 프로토콜로 전송이 가능하다. 또한 새로운 Globus XIO 드라이버가 추가되면 추가된 드라이버의 설정사항들을 모듈화하여 플러그인(plug-in) 방식으로 동적 추가가 가능하도록 설계하였다. <그림 4>는 본 툴의 통신 구조를 나타낸다.



〈 그림 4 〉 DDMG의 통신 구조

지금까지 그리드 환경을 위한 데이터 분산 전송 기법으로 P2P 데이터 전송 기술을 응용한 DDMG 데이터 분산 전송 메커니즘를 제시하였다. 이를 바탕으로 Globus XIO 라이브러리를 활용하여 서버/클라이언트 툴을 구현하였으며, 다음 장에서는 본 구현물의 데이터 전송 성능을 일반 유니캐스트 전송 기법과 비교 분석하였다.

4. DDMG의 성능 측정 및 분석

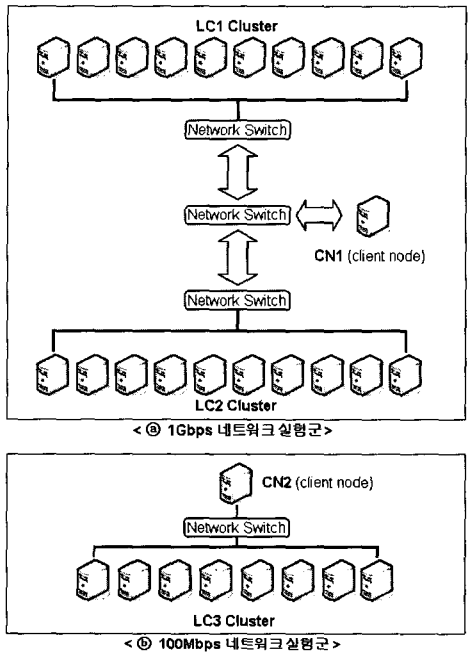
본 장에서는 DDMG와 일반 유니캐스트 전송 기법의 성능을 LAN과 WAN 환경에서 각각 측정하여 비교 분석하였다. LAN 환경에서는 1Gbps 네트워크망을 활용하였으며, WAN 환경은 한국과학기술연구망(KREONET)[24]을 활용하여, 서울, 대전, 광주, 포항 등 총 4개 사이트를 대상으로 실험하였다. 실험은 데이터를 송신할 노드 수에 따라 걸리는 데이터 전송 시간을

DDMG와 일반 유니캐스트 전송 기법에 대해서 측정하는 방법으로 실시하였다. LAN환경에서의 실험은 고속 로컬 네트워크망과 보다 범용적인 저속 로컬 네트워크망에서의 DDMG와 일반 유니캐스트의 전송 성능을 비교 분석하였다. WAN 환경에서의 실험은 송신자가 연구망에 직접 연결되어있을 때의 성능과 송신자가 간접적으로 연구망에 연결되어 있을 때의 성능을 비교 분석하였다. 본 실험은 10번씩 반복 실험하였으며 각 산출값을 평균하여 작성하였다.

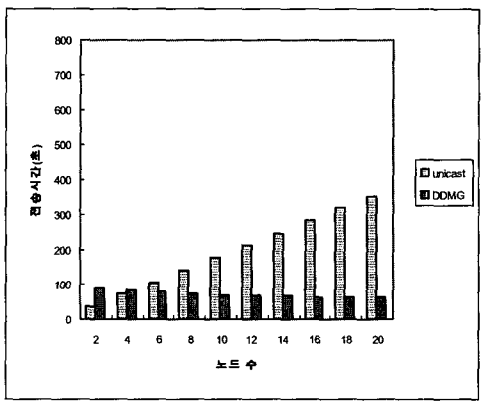
4.1 LAN 환경에서의 성능 측정

LAN 환경에서의 실험을 위해서 <그림 5>과 같이 시스템을 구성하였으며 각 노드의 운영체제는 RedHat Linux 9, 디스크 드라이브는 회전속도가 7200rpm인 E-IDE 인터페이스의 디스크 드라이브를 사용하였다. 고속 로컬 네트워크망 실험(<그림 5>의 ⓐ)을 위해서 1Gbps 네트워크 카드가 장착되어있는 20개의 서버 노드와 1개의 클라이언트 노드로 구성하였다. 또한 저속 로컬 네트워크망 실험(<그림 5>의 ⓑ)을 위해서는 100Mbps 네트워크 카드가 장착되어 있는 9개의 서버 노드와 1개의 클라이언트 노드로 구성하였다. 본 실험은 서버 노드를 추가해가며 각각 서버가 데이터를 전송받는데 걸리는 시간을 측정하는 방법으로 실시하였다.

<그림 6>은 1Gbps 네트워크군에서 1GBytes의 데이터를 전송하는데 걸리는 시간을 DDMG와 일반 유니캐스트 전송기법을 이용하였을 경우를 비교한 그래프이다. 이 그림에서 보는 바와 같이 일반 유니캐스트 방식으로 전송할 때의 전송시간은 노드 수에 비례해서 계속 증가하는 반면, DDMG를 이용하면 전송 시간이 거의 일정하며, 노드수가 증가 할수록 전송시간이 약간씩 짧아지는 결과를 가져온다. 서버 노드 수가 2~4개로 매우 적을 경우에 전송 시간이 일반 유니캐스트 방식의 전송시간보다 높은 이유는, DDMG의 전송 기법의 특성상 디스크의 입출력 부하가 일반 유니캐스트 전송보다 크기 때문이다.(<부록 B> 참조).



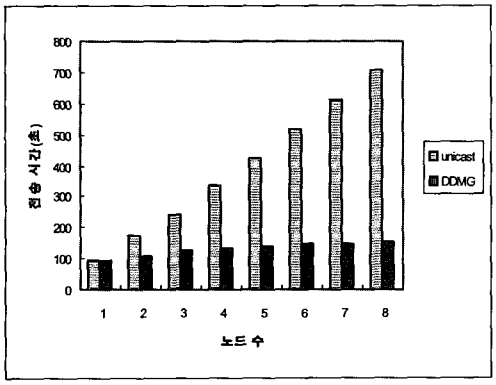
〈 그림 5 〉 LAN 환경의 시스템 구성도



〈그림 6〉 1Gbps LAN 환경에서 노드 수에 따른 전송 시간

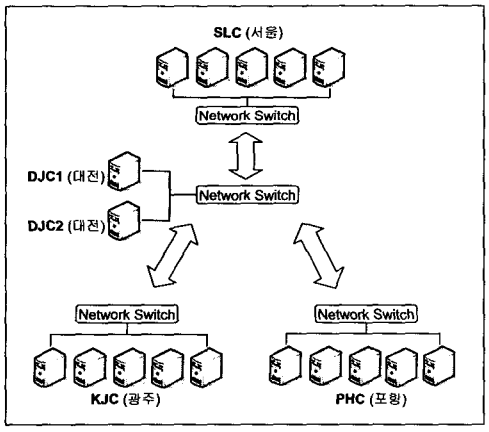
<그림 7>은 100Mbps 네트워크군에서 1G bytes의 데이터를 전송하는데 걸리는 시간을 DDMG와 일반 유니캐스트 전송 기법에 대해

서 비교한 그래프이다. 이 그림에서도 일반 유니캐스트 전송은 <그림 6>과 같이 노드 수에 비례하여 전송 시간이 증가하고 있다. 하지만 네트워크 대역폭이 100Mbps로 제한되어 있기 때문에 노드 수가 소수일 때의 데이터 전송 시간은 유니캐스트 전송과 DDMG 전송이 서로 비슷하다.



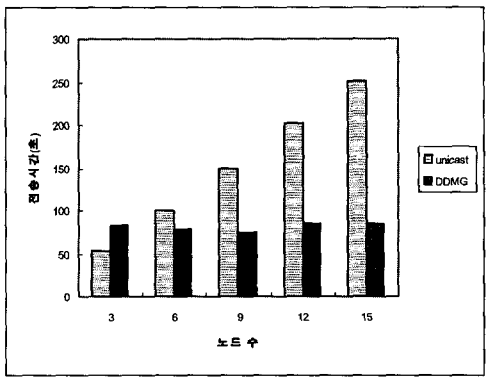
〈그림 7〉 100Mbps LAN환경에서 노드 수에 따른 전송시간

4.2 WAN 환경에서의 성능 측정



〈 그림 8 〉 WAN 환경의 시스템 구성도

WAN 환경에서의 실험은 <그림 8>과 같이 시스템을 구성하였으며, 각 노드의 운영체제는 RedHat Linux 9, 디스크 드라이브는 회전속도 7200rpm인 E-IDE인터페이스의 디스크 드라이브를 사용하였다. 송신 노드가 연구망에 직접적으로 연결되어 있지 않은 DJC2를 제외한 나머지 노드에는 1Gbps 네트워크 카드가 장착되어 있으며 DJC2에는 100Mbps 네트워크 카드가 장착되어 있다. 대전에 위치한 DJC1 및 DJC2 노드를 데이터 송신자로 활용하였으며, 서울, 대전, 광주, 포항에 위치한 노드들을 데이터 수신자로 활용하였다. 성능 측정은 서울, 광주, 포항에 위치한 노드들을 각각 1 노드씩 추가해가며 송신자가 각 수신자로 1GBytes의 데이터를 전송하는데 걸리는 시간을 측정하였다.



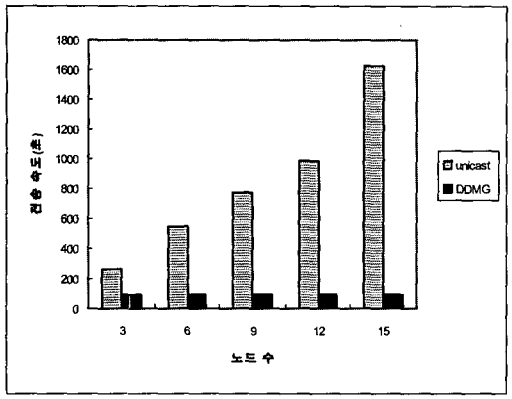
〈그림 9〉 1Gbps WAN 환경에서 노드 수에 따른 전송 시간

<그림 9>는 데이터 송신 노드가 연구망과 직접 연결되어 있을 때의 일반 유니캐스트와 DDMG의 전송 시간를 측정하여 비교한 그래프이다. 이 그림에서 보는 바와 같이 WAN 환경에서도 LAN 환경의 고속 로컬 네트워크망 실험과 같이 노드 수가 적은 경우 디스크 입출력 병목현상으로 인한 DDMG의 성능 저하가 나타난다. 또한 노드 수가 증가 할수록 일반 유니캐스트 전송 시간은 비례적으로 증가하는 반면, DDMG의 전송 시간은 계속 일정하였다.

<그림 10>은 데이터 송신 노드가 연구망과 직접 연결되어 있지 않을 때의 일반 유니캐스

트와 DDMG의 전송시간을 비교한 그래프이다. 이 그림에서 보는 바와 같이 송신 노드의 네트워크 대역폭이 크지 못할 경우(100Mbps)에는 노드의 대역폭이 충분히 확보되었을 때(1Gbps)보다 일반 유니캐스트 전송 시간은 크게 증가하였지만, DDMG 전송의 경우에는 큰 전송 시간의 변화 없이 좋은 성능을 보여준다.

이렇게 LAN 환경과 WAN 환경에서의 성능 실험을 통해서, 데이터를 전송할 노드수가 적고 네트워크 대역폭이 충분히 클 경우에는 일반 유니캐스트 전송 방법이 DDMG 전송 방법보다 좋은 성능을 보여주었다. 하지만 데이터를 전송할 노드 수가 많거나 네트워크 대역폭이 크지 못한 환경에서는 DDMG 전송 방법이 일반적인 유니캐스트 전송 방법보다 전송 시간을 상당히 단축할 수 있음을 보여주고 있다. 본 논문에서 제안한 DDMG 전송 기법은 대용량 데이터의 복제 및 전송이 잦은 고에너지 물리 연구나 기상 예보와 같은 그리드 응용에 활용성이 높을 것으로 예상된다.



〈그림 10〉 WAN 환경에서 노드 수에 따른 전송시간 비교(100Mbps)

5. 결 론

그리드 환경에서 데이터 분산 전송의 요구가 증가하고 있다. 기존의 LAN 환경에서는 데이터 분산 전송을 위해서 멀티캐스트 전송 방법을 널리 활용하였으나, 전송 성능이 일반 유니캐스트 전송보다 좋지 못하고, 하드웨어 지원 없이는 범용적인 WAN 환경에서 사용하기 어렵기 때문에 그리드 환경에서 활용하는 데에 한계가 있었다. 이에 본 논문에서는 네트워크 자원의 효율적인 활용과 그리드 환경에서 다양한 프로토콜을 지원하기 위하여 P2P 데이터 공유 기술을 응용하고 Globus XIO를 활용한 DDMG 전송 기법을 제안 및 구현하였다.

본 논문에서 제안한 틀을 LAN 환경과 WAN 환경에서 노드 수 증가에 따른 전송 시간을 일반 유니캐스트 전송과 비교 측정하였다. 그 결과 유니캐스트 전송은 LAN 환경과 WAN 환경 모두 데이터 수신 노드 수가 증가하면 전송 시간이 노드 수에 비례하여 증가하였다. 또한, 네트워크 대역폭이 1Gbps이상으로 충분히 크며 노드 수가 소수일 경우 유니캐스트 전송 방식이 DDMG 전송에 비하여 더 좋은 전송 성능을 보여주었다. 이에 반해, DDMG 전송은 네트워크 대역폭이 충분히 클 경우 노드 수에 상관없이 계속 일정한 전송 시간을 보여주었으며, 네트워크 대역폭이 작을 경우에는 전송 시간이 증가하였으나, 유니캐스트 전송에 비하여 현저히 작은 증가폭을 보여주었다.

본 논문에서 제안한 DDMG 전송 기법을 실질적으로 활용하기 위해서는 다음과 같은 부분의 개선이 필요하다. DDMG 전송 기법은 P2P의 데이터 공유 기술과 다르게 전송 받을 서버의 수만큼만 전송할 데이터를 같은 크기로 나누게 된다. 데이터를 이렇게 나누게 되면, 데이터를 전송받는 서버중 하나가 다른 서버들에 비하여 전송속도가 현저하게 느릴 경우, 이 느린 서버로 인하여 전체 전송 완료 시간이 늦어지게 된다. 이러한 구조적 단점을 극복하기 위해서 클라이언트에 전송속도 모니터링 기능을 추가하여 전송속도가 느린 노드에게는 보다 적

은 양의 데이터를 전송하고 전송속도가 빠른 노드에게는 보다 많은 데이터를 전송하도록 하는 기능이 필요하다. 또한, 데이터 전송 중에 하나의 노드가 유실될 경우 다른 노드에게도 영향을 주게 되므로 이를 위한 결함 허용(fault tolerent) 기능에 대한 연구도 필요하다. 그리고 P2P 프로토콜의 전형적인 문제인 방화벽과 NAT(Network Address Translator)에 의한 통신 제한 문제에 대한 연구도 필요하다. 이러한 부분들은 반드시 개선되어야 하는 부분들이며 현재 이를 위한 연구가 진행 중이다. 이렇게 개선된 DDMG 전송 기법은 대용량의 데이터 전송이 잦은 고에너지 물리 연구나 기상 연구에 활용 가치가 높을 것으로 예상된다.

Acknowledgement

본 실험을 위해 테스트베드 계정을 제공해주신 한국과학기술정보연구원의 곽재혁 연구원과 함재균 연구원, 그리고 본 논문을 작성하는데 많은 조언을 주신 명훈주 연구원, 정용환 연구원, 김상완 연구원, 권오경 연구원, 안준언 연구원, 그리고 안선일 연구원께 감사드립니다.

참 고 문 헌

[1] The Grid Physics Networks (GriPhyN) project, http://www.griphyn.org

[2] The Large Haron Collider (LHC) Project, http://lhc.web.cern.ch/lhc

[3] The Particle Physics Data Grid Project, http://www.cacr.caltech/edu/ppdg

[4] B. Allcock, J. Bester, J. Bresnahan, A.L. Chervenak, I. Foster, C. Kesselman, S. Meder, V. Nefedova, D. Quesnel, and S. Tuecke, "Data Management and Transfer in High-Performance Computational Grid Environments," *Parallel Computing, Vol. 28, Issue 5, pp. 749-771,* 2002.

[5] A. Chervenak, I. Foster, C. Kesselman, C. Salisbury, and S. Tuecke, "The Data Grid: Towards an Architecture for the distributed Management and Analysis of Large Scientific Datasets," *Journal of Network and Computer Application, Vol. 23, no. 3, pp. 187-200,* 2000.

[6] I. Foster, C. Kesselman, and S. Tuecke, "the Anatomy of the Grid: Enabling Scalable Virtual Organizations," *International Journal of Supercomputer Applications, Vol. 15, no. 3,* 2001.

[7] W. Allcock, J. Bseter, J. Bresnahan, S. Meder, P. Plaszczak, and S. Tuecke, "GridFTP: Protocol Extensions to FTP for the Grid," *GFD-R.020,* January 2004.

[8] B. C. Neuman, Theodore Ts'o, "Kerberos: An Authentication Service for Computer Networks," *IEEE Communications, Vol, 32, Issue 9, pp. 33-38.* September 1994.

[9] V. Welch, F. Siebenlist, I. Foster, J. Bresnahan, K Czajkowski, J. Gawor, C. Kesselman, S. Meder, L. Pearlman, and S. Tuecke, "Security for Grid services," *High Performance Distributed Computing Proceedings, pp. 48-57,* June 2003.

[10] EGEE project, http://eu-egee.org

[11] TeraGrid project, http://teragrid.org

[12] S. Pickels et al, "The TeraGyroid Project," *Proceedings of the 10th Globual Grid Forum(GGF10) : Workshop on Case Studies on Grid applications,* March 2004.

[13] W. Allcock, J. Bresnahan, R. Kettimuthu, and J. Link, "The Globus eXtensible Input/Output System (XIO): A Protocol Independent IO System for the Grid,"

19th IEEE International Parallel and Distributed Processing Symposium (IPDPS '05) - Workshop 4, pp. 179, 2005.

[14] K. Jeacle and J. Crowcroft, "Reliable high-speed Grid data delivery using IP multicast," *Proceedings of All Hands Meeting*, 2003.

[15] A. Dunkels, "Minimal TCP/IP implementaition with proxy support," *Technical Report SICS-T-2001/20-SE, Swedish Institute of Computer Science,* Feburary 2001.

[16] J. P. Macker, "The Multicast Dissemination Protocol (MDP) Toolkit," *Proceedings of IEEE MILCOM, Vol. 1, pp. 626-630,* 1999.

[17] B. Adamson, C. Bormann, M. Handley, and J. Macker, "NACK-Oriented Reliable Multicast Protocol (NORM)," *RMT Working Group Internet Draft*, January 2004.

[18] M. Hofmann, "Scalable multicast communication in wide area networks," *PhD Thesis, University of Karlsruhe, Germany*, 1998.

[19] M. P. Barcellos, M. Nekovee, M. Daw, J. Brooke, and S. Olafsson, "Reliable Multicast for the Grid: a comparison of protocol implementations," *e-Science All-Hands Meeting, Nottingham, UK*, September, 2004.

[20] H. Eriksson, "MBONE : The multicast backbone," *Communication of ACM*, 1994.

[21] D. Plonka. "Napster traffic measurement," http::/net.doit.wisc.edu/data/Napster, March 2000.

[22] S. Saroiu, K. P. Gummadi, R. J. Dunn, S. D. Gribble, and H. M. Levy, "An analysis of internet content delivery systems," *Proceedings of the Fifth Symposium on Operating Systems Design and Implementation (OSDI 2002), pp. 315-328,* December 2002.

[23] http://www-unix.mcs.anl.gov/~kettimut/xio/XIO_Perf ormance.pdf

[24] http://kreonet.re.kr

부록 A : DDMG의 서버/클라이언트 유사 부호(Pseudo Code)

〈표 A-1〉 클라이언트의 유사부호(Pseudo Code)

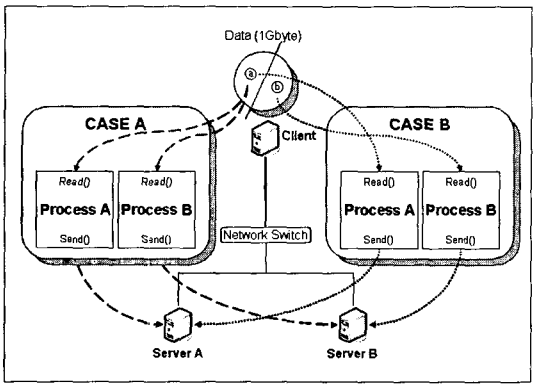
```
사용자로부터,
 - 전송할 파일명
 - 서버 주소
 - 각 서버의 프로토콜 명
을 입력 받음;

for(각각의 서버) {
 서버에 접속하여 파일명과 서버 수를 전송함;
}

전송할 데이터의 크기를 서버 수만큼 나눔;

위 크기를 바탕으로 각 서버에게 전송할 데이터의
시작 주소와 끝 주소를 저장;

for(각각의 서버) {
 서버로부터 접속할 주소 정보를 받음;
}

for(각각의 서버) {
 서버에게,
 - 이웃 서버들로부터 받은 주소 정보
 - 이웃 서버들의 프로토콜 명
 - 서버가 전송할 데이터의 시작 주소
 - 서버가 전송할 데이터의 끝 주소
 정보를 전송함;
}

for(각각의 서버) {
 서버로부터 전송 준비 완료 메시지를 받음;
 서버로 데이터 전송을 시작;
}
```

〈표 A-2〉 서버의 유사부호(Pseudo Code)

```
클라이언트로부터,
 - 파일명
 - 서버의 수
정보를 전송 받음;

for(서버의 수) {
 새로운 통신 소켓을 초기화 함;
}

클라이언트에게 초기화된 통신 소켓 정보를 보냄;

서버로부터,
 - 이웃 서버들의 주소
 - 이웃 서버들의 프로토콜 명
 - 전송받을 데이터의 시작 주소
 - 전송받을 데이터의 끝 주소
정보를 전송 받음;

전송받을 파일을 초기화 함;

for(각각의 이웃 서버) {
 서버의 주소로 접속;
 전송받을 데이터의 시작 주소와 끝 주소 전송;
}

for(각각의 통신 소켓) {
 이웃 서버로부터,
 - 전송받을 데이터의 시작 주소
 - 전송받을 데이터의 끝 주소
 정보를 전송받음;
 데이터를 전송 받을 준비를 함;
}

클라이언트에게 전송준비 완료 메시지 보냄;

클라이언트로부터 데이터를 전송 받음;

for(각각의 이웃서버) {
 전송받은 데이터를 다른 서버에게 전송;
}
```

부록 B : DDMG 전송 기법의 디스크 입출력 부하 실험

본 실험은 DDMG 전송에 걸리는 디스크 입출력 부하를 측정하기 위한 실험이다. 이를 위하여 <그림 B-1>과 같이 3개의 노드로 시스템을 구성 하였다. 각 노드들의 운영체제는 Redhat Linux 9이며, 1Gbps 네트워크 카드가 장착되어있으며, 디스크 드라이브는 회전속도 7200rpm인 E-IDE인터페이스의 디스크 드라이브를 사용하여 <표 B-1>과 같은 입출력 성능을 지니고 있다. 또한 이 노드들은 1Gbps 로컬 네트워크망에 있다.



〈그림 B-1〉 디스크 입출력 부하 측정을 위한 시스템 구성

〈표 B-1〉 클라이언트 노드의 입출력 성능

소 스	읽는 속도(MB/s)
버퍼메모리에서 읽는 속도	673MB/s
디스크에서 읽는 속도	53MB/s

본 실험에서는 <그림 B-1>에서 보는 바와 같이 하나의 클라이언트에서 같은 크기로 나눠진 데이터를 각각 server A와 server B에 전송하는데 걸리는 시간을 측정하였다. 이 과정에서 다음과 같은 2가지의 경우를 비교하였다.

- CASE A : 하나의 데이터를 2부분으로 나눠 같은 부분을 동시에 2개 서버에 전송하는데 걸리는 시간 측정
- CASE B : 하나의 데이터를 2부분으로 나눠 서로다른 부분을 동시에 2개 서버에 전송하는데 걸리는 시간 측정

즉, CASE A와 CASE B의 차이점은 CASE A에서는 나눠진 데이터의 같은 부분을 각 서버에 전송하는데 반하여, CASE B에서는 나눠진 데이터의 각기 다른 부분을 각 서버에 전송한다는 점이다. 이러한 실험을 통하여 서로 다른 데이터를 읽는데 걸리는 디스크 입출력 부하를 비교 측정해 보았다.

본 실험의 결과는 <표 B-2>와 같다. 즉, 본 실험에서는 같은 디스크 주소에 위치한 데이터를 전송하는데 걸리는 시간(CASE A)이 다른 디스크 주소에 위치한 데이터를 전송하는데 걸리는 시간(CASE B)의 3.5배 더 빠르다는 것을 알 수 있다.

〈표 B-2〉 500Mbytes의 데이터를 전송하는데 걸리는 시간

실험군	전송 시간(초)
CASE A	24.2
CASE B	85.3

DDMG의 전송 기법은 CASE B의 전송 기법을 활용 확장하였기 때문에, 네트워크 대역폭이 충분히 큰 경우에는 디스크 입출력 병목 현상으로 일반 유니캐스트 전송보다 느린 전송 성능을 갖게 된다.

◐ 저 자 소 개 ◑

김 형 진 (Hyungjinn Kim)

2004년 한국항공대학교 기계과 졸업(학사)
2005년~현재 과학기술연합대학원 대학교 그리드 컴퓨팅전공 석사과정. 관심 분야는 그리드 컴퓨팅, 운영체제 구조론.
E-mail: qanii@kisti.re.kr

이 종 숙 (Jongsuk Ruth Lee)

1992년 충남대학교 전산학과 석사
2001년 University of Canterbury (New Zeland) 전산학과 공학 박사
1992년~1993년 한국 전자통신연구원 연구원
1999년~2002년 University of Canterbury (New Zeland) 연구원
2002년~현재 한국과학기술정보연구원 선임연구원
2005년~현재 과학기술연합대학원대학교(UST) 그리드 슈퍼컴퓨팅 전공부분 겸임교수
E-mail: jsruthlee@kisti.re.kr