병렬 연결 간의 트래픽 간섭 현상 분석 및 대역폭 예측☆

Analysis of the Interference between Parallel Socket Connections and Prediction of the Bandwidth

김 영 신* 허 의 남** 김 일 중*** 황 준 ****
Young-Shin Kim Eui-Nam Huh Il-Jung Kim Jun Hwang

요 약

최근 들어, 고에너지 물리학 등의 과학 기술이 고도로 발전함에 따라 실험 데이터 등이 대량으로 생산되고 있다. 따라서 이러한 데이터를 분산된 스토리지를 이용하여 저장하고 있으며, 이로 인해 지역적으로 분산된 자원을 하나의 자원처럼 사용할 수 있는 환경이 요구되고 있다. 그러나 분산된 자원을 관리하는 대부분의 시스템은 다른 관리 시스템과 호환이 되지 않는다는 문제점을 가지고 있다. 데이터 그리드는 이러한 문제를 해결하고 있으며, 효율적인 데이터 관리를 위해 데이터 전송의 안전성과 신속성을 보장하는 GridFTP를 이용하고 있다. 그러나 이러한 툴의 사용은 병렬 전송 기술을 이용하기 때문에 네트워크 과부하가 발생할 가능성이 높으며, 많은 소켓연결을 생성한 응용 프로그램이 네트워크 대역폭의 대부분을 점유하는 문제점이 발생한다. 그러나 현재 병렬 전송 기술의 특성이나 문제점에 대한 분석이 이루어지지 않고 있으며, 특히, 소켓 연결들 간의 간섭 현상에 대한 분석이 이루어지지 않고 있다. 따라서 본 논문에서는 병렬 전송의 특성을 여러 실험 결과를 통하여 분석하고, 소켓 연결들 간의 간섭 현상을 분석한다. 응용 프로그램이 요구하는 네트워크 대역폭 확보를 위한 소켓 연결의 개수 계산과 새로운 소켓 연결에 의한 간섭 발생 이후 기존의 응용프로그램이 사용할 대역폭을 예측한다.

Abstract

Recently, many researchers have been studied several high performance data transmission techniques such as TCP buffer Tuning, XCP and Parallel Sockets. The Parallel Sockets is an application level library for parallel data transfer, while TCP tuning, XCP and DRS are developed on kernel level. However, parallel socket is not analyzed in detail yet and need more enhancements. In this paper, we verify performance of parallel transfer technique through several experiments and analyze character of traffic interference among socket connections. In order to enhance parallel transfer management mechanism, we predict the number of socket connections to obtain SLA of the network resource and at the same time, affected network bandwidth of existing connections is measured mathematically due to the interference of other parallel transmission. Our analytical scheme predicts very well network bandwidth for applications using the parallel socket only with 8% error.

☞ Keyword : TCP buffer tuning, ATBT, DRS, Parallel Trasnfer

1. 서 론

최근 들어, 고에너지 물리학, 생명공학, 나노물질 등의 과학 기술이 고도로 발전함에 따라 실험 데이터나 수집된 데이터 등이 대량으로 생산되고 있다. 그러나 이러한 대량의 데이터를 특정한 사용자가 독점적으로 보유하고 관리하는 것은 거의 불가능한 일이다. 또한 최근 연구 환경이 여러 분야를 협업하는 경향으로 조성됨으로써, 지역적으로 분산된 자원을 하나의 자원처럼 사용할 수 있는 환경이 요구되고 있다.

그러나 분산된 자원을 관리하는 대부분의 시스템은 다른 관리 시스템과 호환이 되지 않는다는 문제점을 가지고 있어, 지리적으로 분산되어 있는 연구기관 간의 협업이 어려울 뿐만 아니라 스토리지 용량 한계

* 준 회 원 : 서울여자대학교 대학원 박사과정
amary46@swu.ac.kr(제1저자)
** 종신회원 : 경희대학교 컴퓨터공학과 조교수
johnhuh@khu.ac.kr
*** 준 회 원 : 한국전산원 차세대인터넷팀 근무
ijkim@nca.or.kr
**** 종신회원 : 서울여자대학교 정보통신공학부 교수
hjun@swu.ac.kr
[2005/06/01 투고 - 2005/06/20 1차 심사 - 2005/11/08
2차 심사 2005/11/15 심사완료]
☆ 본 연구는 한국전산원에서 지원됨

로 인한 분산 저장도 어렵다고 할 수 있다. 이러한 문제점을 해결하기 위해 데이터 그리드가 개발되었으며, 이로 인해 전 세계의 분산된 데이터를 용이하게 통합, 관리, 분석할 수 있게 되었고, 데이터의 저장 및 접근, 사용자의 편의성과 대량 데이터의 신속, 정확한 전송 등의 서비스가 가능하게 되었다[1].

Globus Alliance Team에서는 그리드 환경에서 분산된 데이터를 안전하고, 빠르게 전송할 수 있도록 GridFTP를 개발하였다[2]. 이 툴은 병렬 전송 기술을 사용함으로써 응용레벨에서 고성능의 전송이 가능하다. 그러나 그리드 환경에서 GridFTP와 같은 응용프로그램의 사용이 증가하게 되면, 네트워크 과부하 발생 가능성이 높아지며, 또한 다른 응용 프로그램에 비해 다수의 소켓 연결을 설정한 응용 프로그램이 네트워크 대역폭의 대부분을 할당받는 문제점이 발생하게 된다. 병렬전송 기술을 효율적으로 사용하기 위해서는 이러한 문제점에 대한 분석이 필수적이라 할 수 있다. 그러나 현재까지는 이에 대한 연구가 이루어지지 않고 있다.

따라서 본 논문에서는 병렬 전송의 효과를 실험을 통하여 알아보고, 소켓 연결들 간의 간섭 현상을 실험 결과를 통하여 분석한다. 더 나아가 호스트 간의 병렬 전송을 효율적으로 관리하기 위해, 네트워크 대역폭 확보를 위한 소켓 연결의 개수 예측과 기존에 응용프로그램이 사용하던 대역폭의 변화를 예측한다.

본 논문은 다음과 같이 구성되어 있다. 2장에서는 각종 데이터 전송 성능 향상 기술들에 대해 알아보고, 3장에서는 초고속망에서 실제 병렬로 데이터를 전송할 경우 성능의 향상 정도, 그리고 병렬 전송의 간섭 현상을 분석하기 위해 여러 가지 실험을 수행하였다. 4장에서는 병렬 전송 관리를 위한 대역폭 예측 모델링과 대역폭 획득을 위한 소켓 연결 개수 계산 모델링을 수행한다. 마지막으로 5장에서는 결론과 향후 연구 과제를 논의한다.

2. 관련 연구

2.1 데이터 전송 제어 알고리즘 수정

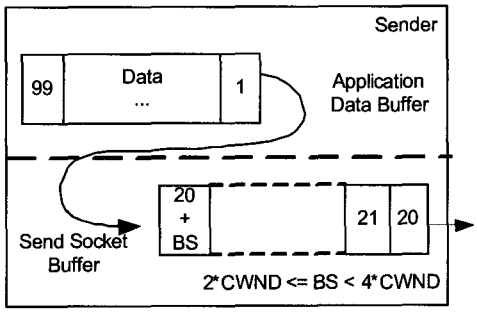
이 방법은 커널 레벨에서 데이터 전송 제어 알고리즘을 수정하는 방법이다. 예를 들어, eXplicit Congestion control Protocol(XCP)과 같은 경우, 패킷의 헤더에 congestion header를 추가하고 이를 이용하여 송신자에게 혼잡을 판단할 수 있는 feedback을 제공함으로써, 혼잡의 발생을 정확하게 판단할 수 있도록 한다[7]. 따라서 판단오류로 전송되는 데이터 량을 줄임으로써 데이터 전송 성능을 향상시킨다. 그러나 이 방법은 congestion header를 추가하기 위해 커널 레벨의 소스를 수정해야만 하므로, 다른 시스템의 커널과 호환되지 않는다는 점과 알고리즘 수정이 어렵다는 문제점을 가진다[7,8].

2.2 TCP buffer tuning

이 기술은 TCP Buffer 크기를 적당하게 조절함으로써 데이터 전송 성능을 향상시키는 기술이다. 여기서 TCP buffer의 적당한 크기는 BDP(Bandwidth Delay Product=bottleneck bandwidth*RTT)를 이용하여 얻을 수 있으며, BDP에 따라 wmem_max 변수를 수정함으로써 전송 성능을 향상시킨다. 그러나 한번 조절한 TCP 버퍼의 크기는 고정되어 있으므로 네트워크 상태 변화가 심할 경우에는 성능 향상을 기대하기 어렵다. 또한 TCP 버퍼 크기를 조절하기 위해서는 전문지식을 갖춘 전문가를 필요로 하기 때문에 일반 사용자가 사용하기에는 어렵다는 문제점을 가지고 있다. 따라서 이러한 문제점을 해결하기 위해 최근에는 Automatic TCP Buffer Tuning(ATBT), Dynamic Right Sizing (DRS)과 같은 Auto Tuning 방법이 연구되고 있다[5,6,8].

2.2.1 ATBT(Automatic TCP Buffer Tuning)

이 기술은 송신 버퍼의 크기를 네트워크의 상태에 따라 자동으로 조절함으로써 성능을 향상시키도록 설계된 기술이다. TCP에서 송신 버퍼의 크기는 Congestion Window (CWND)의 영향을 받는데, ATBT에서는 2*CWND보다 같거나 크고, 4*CWND보다는 작은 범위에서 버퍼의 크기가 결정된다. 만약, 이 범위를 벗어날 경우 송신 버퍼의 크기는 재



〈그림 1〉 송신측 버퍼의 자동 튜닝

조정된다.

그림 1은 ATBT의 auto tuning을 나타내는 것으로, 응용 레벨에서 데이터(1~99)를 소켓 버퍼로 이동하고 이를 소켓 버퍼에서 네트워크로 전송하는 상태를 나타내고 있다. 1~99의 데이터 중 19 데이터까지는 전송된 상태이며, 20부터 소켓 버퍼에 대기 중인 상태를 나타내고 있다. 소켓 버퍼의 크기는 현재의 CWND에 의해 영향을 받아 결정되며, '20+BS'만큼의 데이터는 응용레벨 데이터 버퍼로부터 소켓버퍼로 이동된다.

그러나 총 소켓 연결들이 요구하는 버퍼 크기가 송신 버퍼의 크기보다 작을 경우에는 fair share algorithm에 의해 버퍼가 할당된다. 각 소켓 연결들에 버퍼를 공평하게 할당하고, 사용되지 않는 버퍼는 'Shared Pool'에 반환한다. 이에 반해 모든 소켓 연결들이 요구하는 버퍼가 송신 버퍼의 크기보다 작을 경우에는'Shared Pool'에 반환되어 있는 메모리를 공평하게 재할당 받는다. 이러한 과정은 tcp_slowtime()에 의해 주기적으로 수행되므로 TCP 연결이 지속되는 동안 동적으로 버퍼 크기가 조절된다.

2.2.2 DRS

DRS 역시 소켓 연결이 설정되어 있는 동안 수신 버퍼의 크기를 자동으로 조절함으로써 성능을 향상시키는 기술이다. TCP는 Effective Window (EWND) 단위로 데이터를 전송하도록 설계되어 있다. 여기서, EWND는 flow control window (FWND)와 congestion control window (CWND) 중 최소값으로 정해지는데, FWND는 수신측 버퍼의 빈 공간에 따라 값이 결정되며, CWND는 현재 네트워크의 혼잡에 정도에 따라 결정된다. 그러므로 CWND와 FWND의 값이 동일할 때, 효율적인 메모리 사용과 최대 네트워크 이용률을 얻을 수 있다.

전반적으로 FWND의 값은 거의 변화가 없으며, 현재 대부분의 운영체제에서 64KB로 설정되어 있어, 초고속 네트워크에서는 EWND가 FWND에 의해 결정될 가능성이 높다. 따라서 수신측에서 RTT를 측정하여 수신 버퍼 크기를 조절함으로써 송신측의 EWND 값을 보다 적절한 값으로 결정되도록 유도한다.

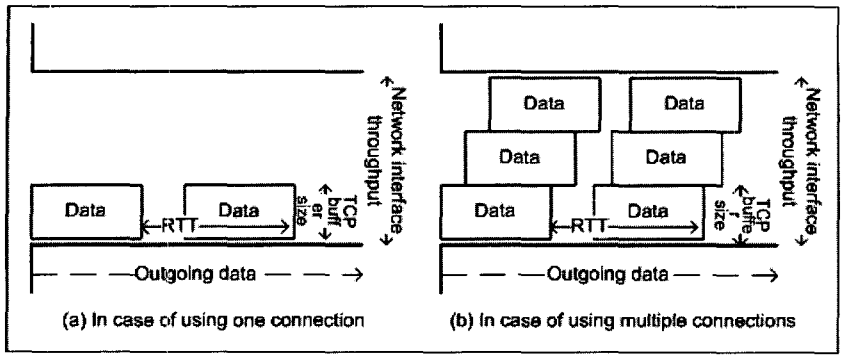
수신측에서의 RTT 측정은 ack를 보낸 후 다음 데이터 도착 시간을 이용하여 결정한다. RTT 측정 간격이 짧아지면 오버헤드가 발생되며, 측정 간격이 길어지면 네트워크 상태의 변화가 심한 상황에서는 부정확한 버퍼 크기가 결정될 가능성이 높다. 따라서 RTT 측정 간격 적정하게 선택되어야 한다.

2.3 Parallel Transfer

이 기술은 송신측과 수신측 간에 다중 소켓 연결을 설정하고, 설정된 소켓 연결들의 개수만큼 분할된 데이터를 각각의 소켓 연결을 통해 동시에 전송하는 기술이다. 이 기술은 응용레벨에서 운영됨으로써, TCP buffer tuning의 단점을 보완함과 동시에 고성능의 원거리 네트워크 환경에서 최대 성능을 발휘할 수 있다. 이러한 장점 때문에 GridFTP에서도 이 기술을 활용하고 있다. GridFTP 프로토콜은 RF959와 다른 IETF 문서들에 의해 정의된 FTP 프로토콜에서 확장된 것으로, 현재는 GridFTP v.2가 발표(2004년)되어 있다[2].

그림 2는 병렬 전송을 나타내고 있다. 병렬 소켓 연결을 이용하여 데이터를 전송하는 상태와 단일 소켓 연결을 이용하여 데이터를 전송하는 상태를 비교하여 보여주고 있다[3][4].

그림 2의 (a)는 단일 소켓 연결을 이용하여 데이터를 전송하는 상태를 나타낸 것으로 네트워크 대역폭, 즉, 네트워크 인터페이스 처리량을 충분히 활용하지 못하고 있다. 그러나 (b)는 여러 개의 소켓

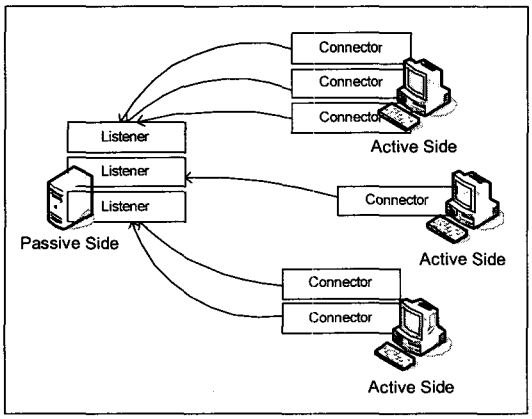


〈그림 2〉 하나의 connection과 다중 connection의 비교

연결을 이용하여 네트워크 대역폭을 최대로 활용하는 모습을 볼 수 있다. 이처럼 병렬 전송 기술은 TCP 버퍼 크기를 조절하지 않았음에도 불구하고 네트워크 대역폭을 최대로 사용할 수 있다.

Striping은 여러 호스트에서 병렬로 한 데이터를 전송하기 위한 기술이다[3][4]. 위의 병렬 전송과 striping 간의 중요한 차이점은 passive side(연결요청을 기다리는 호스트)에서 listener(server)들이 연결 요청을 기다릴 때, 하나의 주소 대신에 주소 배열을 가진다는 것이며, 또한 각 주소들은 여러 호스트에 분산되어 있을 수 있다. 따라서 이 기술은 하나의 데이터가 여러 호스트에 분산되어 저장된 경우 각 호스트에 소켓 연결을 생성하여 분산되어 있는 데이터를 전송받을 수 있다.

그림 3은 병렬 전송과 striping의 통합을 나타내



〈그림 3〉 Parallelism 과 Striping 의 통합

고 있다. 이런 경우 stripe은 passive side의 listener들 중에 하나와 형성된 소켓 연결들의 집합을 의미한다. 한 stripe은 여러 소켓 연결들로 구성될 수 있다[3,4]. 병렬 전송의 경우에는 성능 향상에 네트워크 인터페이스 카드의 제한을 받지만, striping 같은 경우에는 이러한 제한을 받지 않고 성능을 향상시킬 수 있다.

그러나 여러 응용 프로그램이 이 기술을 이용하여 다수의 소켓 연결을 생성하고 데이터를 전송할 경우, 각 소켓 연결의 성능은 서로에게 영향을 주게 된다. 또한 다른 응용프로그램보다 더 많은 소켓 연결들을 생성하는 응용 프로그램이 네트워크 대역폭의 대부분을 점유하는 문제점이 발생된다.

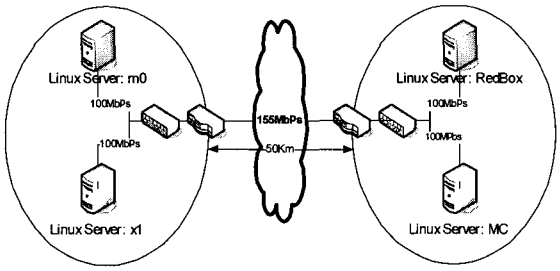
3. 실험

3.1 실험 환경

아래 그림 4와 같이, KOREN[1] 155M 백본 망과 100Mbps fast ethernet을 사용하는 네트워크에 4대의 컴퓨터를 연결하고, 이러한 환경에서 여러 실험을 수행하였다.

m0와 x1은 A지역에 위치해 있으며, RedBox와 MC는 B지역에 위치해 있다. A지역과 B지역 간의 거리는 약 50Km이다. B지역에 위치해 있는 호스

1) 광대역통합 연구개발망 (KOREN)은 첨단 분야 연구를 수행할 수 있도록 지원하는 광대역, 고품질의 네트워크를 말함



〈그림 4〉 실험 환경

트, RedBox와 MC의 성능은 Pentium III 800MHz 이며, B지역에 위치해 있는 두 호스트 중, m0는 Pentium III 650MHz이고, x1의 성능은 Pentium III 733MHz 이다.

또한, 송신측 호스트의 성능에 따라 데이터 전송률의 변화를 알아보기 위해 m0(Pentium III 650MHz) 호스트 대신 Pentium IV 1.53GHz 성능의 호스트를 실험 중간에 대체 사용하였다. x1은 주로 A지역과 B지역 간에 트래픽을 발생시키는데 사용되었다.

3.2 병렬 전송에 의한 성능 향상 실험

병렬 소켓 연결을 이용하여 데이터를 전송할 경우 데이터 전송 성능의 향상 정도를 알아보기 위해 실험용 병렬 전송 툴을 구현하였다. 이 툴은 Grid FTP1.0 소스를 분석한 후, 그 내용을 토대로 구현하였다. 대용량 데이터(500MB)를 단일, 2개, 4개, 8개, 16개, 32개의 소켓 연결(이하 SCi로 함-i는 소켓 연결의 개수를 나타냄)들을 이용하여 전송하고, 각각의 경우에 수신측에서 데이터 도착 시간을 측정하였다. 또한 네트워크 부하나 호스트의 성능이 병렬 전송에 영향을 미치는지 알아보기 위해 다음과 같은 여러 실험을 수행하였다.

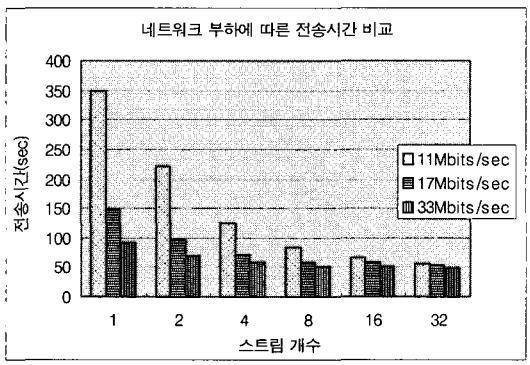
(1) 네트워크 부하에 따른 데이터 전송 성능의 변화

본 실험은 네트워크 부하가 데이터 전송 성능에 미치는 영향을 알아보기 위해 네트워크 상태를 달리하여 데이터를 전송시키고 전송 시간을 측정하였다. 네트워크에 부하를 발생시키기 위해 x1과 RedBox에서는 스트림 개수를 증가하면서 더미패킷을 전송한다. 이렇게 부하를 조절함으로써 가용 대역폭을 약 11Mbits/sec, 17Mbits/sec, 33Mbits/sec로 획득한다. 각 네트워크 부하 상태에서 m0에서 MC로 SC_1을 이용하여 데이터를 전송하고, 도착 시간을 측정하였다. 또한, 이와 동일한 과정으로 SC_2, SC_4, SC_8, SC_{16}, SC_{32} 을 이용하여 실험을 수행하였다.

그림 5는 본 실험의 결과를 나타내고 있다. SC_2 보다 SC_{32}에 가까워질수록 데이터 도착 시간이 단축되고 있음을 알 수 있다. 또한, 가용 대역폭이 11Mbits/sec 인 경우, 즉, 네트워크 부하가 높은 경우에 데이터 전송 시간이 상대적으로 크게 단축되고 있음을 알 수 있다.

만약, 대역폭이 100Mbits/sec인 네트워크에서 대역폭을 90% 활용할 수 있다고 가정하면, 500MB의 데이터를 전송하는데 소요되는 시간이 약 45초로 계산된다. 그런데 그림 5의 그래프를 보면, SC_{32}로 데이터를 전송할 경우, 데이터 도착 시간은 약 48~56초로 측정되었으므로, 네트워크 대역폭 대부분이 활용되고 있음을 알 수 있다. 즉, 다른 응용 프로그램이 사용하던 대역폭까지 사용하고 있음을 알 수 있다.

또한, 가용 대역폭이 작아질수록 전송 시간이 단축되는 폭이 커지는 현상을 볼 수 있는데, 이는 병렬 전송이 네트워크에 부하가 클수록 큰 효과를 얻을 수 있음을 의미한다.



〈그림 5〉 네트워크 부하가 다른 경우마다 채널수에 따른 데이터 전송 시간 측정

(2) 호스트 성능에 따른 데이터 전송 성능의 변화

송신 호스트의 성능이 병렬 전송에 미치는 영향을 알아보기 위해 본 실험을 수행하였다. 성능이 다른 두 호스트에서 같은 크기의 데이터를 각각 전송하고 전송 시간을 측정하였다. A지역에 위치한 Pentium III 800MHz 성능의 호스트에서 B지역으로 데이터를 송신하고 데이터 도착시간을 측정하였다. 또한, Pentium IV 1.53GHz 성능의 호스트에서도 B지역으로 데이터를 전송하고 도착시간을 측정하였다.

본 실험은 네트워크 가용 대역폭이 11Mbits/sec인 상태에서 수행되었고, 실험 결과가 그림 6에 나타나 있다.

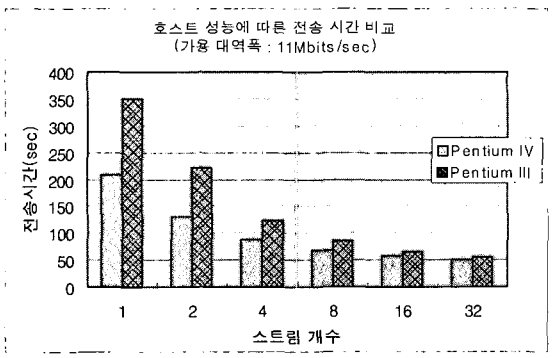
Pentium IV와 Pentium III에서 전송한 데이터의 도착 시간이 SC_1에는 약 150초 정도 차이를 보이다가 SC_{32}에 가까워질수록 차이가 적어지는 것을 볼 수 있다. 이는 다수의 소켓 연결을 이용하여 데이터를 전송 할 경우, 성능이 낮은 호스트도 성능이 높은 호스트와 비슷한 전송 효과를 얻을 수 있음을 의미한다.

3.3 병렬 스트림 간의 간섭 현상

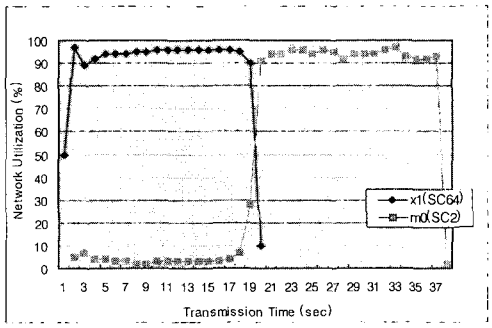
응용 프로그램에서 생성한 병렬 소켓 연결 간의 간섭을 분석하기 위해, x1과 MC, m0와 RedBox 간에 병렬 소켓 연결을 설정하고, 200MB 데이터를 전송하였다. 먼저, x1에서 데이터 전송이 시작하고, 뒤를 이어 m0에서 데이터 전송을 시작하였다. 각 실험마다 x1은 SC_{64} 로 데이터를 전송시켰고, m0은 SC_2에서 SC_{64}로 소켓 연결 수를 변경시켜 데이터를 전송하였다. 이러한 과정에서 스니퍼를 이용하여 각 응용 프로그램들이 사용하는 네트워크 이용률을 측정하였다. 그림 7~10은 각각의 실험 결과를 나타내고 있다.

그림 7의 그래프는 m0에서 SC_2를 이용하여 데이터를 전송할 경우, x1이 데이터를 전송할 때 받는 영향을 알아보기 위해 수행한 실험 결과를 나타내고 있다. 실험 결과를 보면, x1은 네트워크의 평균 90.16%를 이용하였고, m0는 평균 9.53% 정도를 이용하는 것으로 나타났다. 두 호스트가 데이터 전송을 시작한 후 약 20초 동안 x1이 높은 이용률을 기록하며 200MB의 데이터를 전송하는 상태를 나타냈다. 그러나 이에 반해, SC_2를 이용하는 m0은 네트워크 대역폭을 거의 이용하지 못하는 상황을 나타냈다. 그러나 20초 이후, 즉, x1이 전송을 마친 후에는 m0의 네트워크 이용률이 증가하여 평균 92%를 기록하였다. 네트워크 대역폭의 대부분을 점유하던 x1의 응용 프로그램이 종료됨에 따라 m0가 대역폭을 활용할 수 있는 상태가 되었기 때문에 나타나는 현상이라 볼 수 있다. 결국, 이번 실험의 결과에서는 x1이 m0의 간섭을 거의 받고 있지 않음을 알 수 있다.

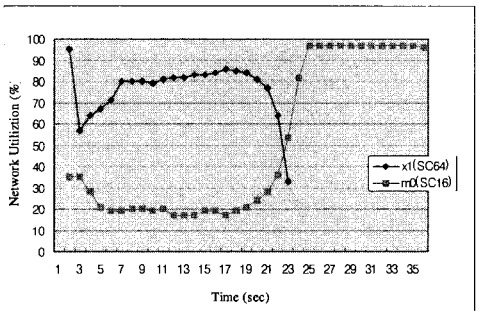
그림 8의 그래프는 m0에서 SC_{16}으로 데이터를 전송할 경우 x1의 데이터 전송 성능에 미치는 영향을 알아보기 위해 수행한 실험의 결과를 나타내고 있다. 데이터 전송 시작부터 전송이 끝나는 24



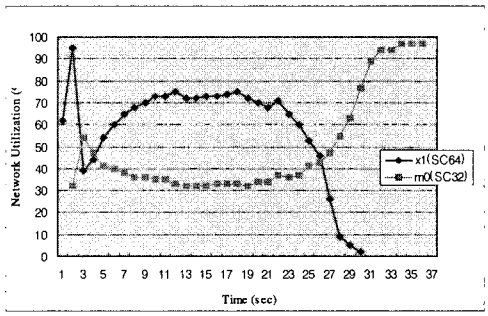
〈그림 6〉 호스트 성능에 따른 병렬 전송 성능 비교



〈그림 7〉 두 호스트가 각각 64개와 2개 스트림을 이용하여 데이터 전송 시 간섭 현상



〈그림 8〉 두 호스트가 각각 64개와 16개 스트림을 이용하여 데이터 전송 시 간섭 현상



〈그림 9〉 두 호스트가 각각 64개와 32개 스트림을 이용하여 데이터 전송 시 간섭 현상

초 지점까지 x1이 사용하는 네트워크 이용률은 평균 76.27%를 기록했으며, m0은 평균 23.82%를 이용하는 것으로 기록되었다. 이번 실험 결과를 보면, 그림 7의 x1이 90.16% 대역폭을 이용했던 것에 비해, 이용률이 대폭 축소된 것을 알 수 있는데, 이는 m0의 SC_{16}이 SC_2보다 많은 대역폭을 할당받아 사용함을 의미한다.

그림 8의 24초 지점에서 두 호스트의 네트워크 이용률이 서로 곡선을 이루면서 교차되는 현상을 볼 수 있다. 이는 데이터 전송을 위해 생성된 쓰레드들이 시간차를 두고 종료하기 때문에 나타나는 현상이라 분석할 수 있다. 즉, x1에서 쓰레드가 순차적으로 종료되면서 확보했던 네트워크 대역폭을 반환하게 되고, m0은 반환된 네트워크 대역폭을 이용함으로써 이용률이 곡선을 그리면서 증가하게 된다.

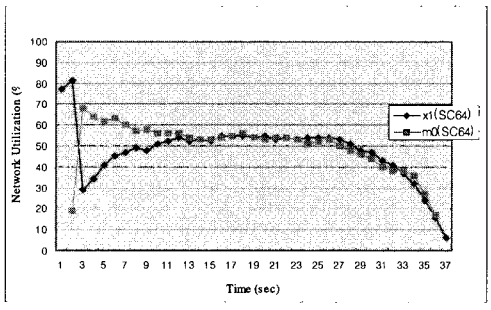
그림 9는 m0에서 SC_{32}로 데이터를 전송할 때, SC_{64}를 이용하는 x1과 m0의 네트워크 이용률을 측정한 결과를 나타내고 있다. 이번 실험에서 x1은 58.69%의 이용률을, m0는 39.33%의 이용률을 기록하였다. 그림 8의 실험과 마찬가지로 m0의 소켓 연결 개수가 증가함에 따라 x1의 네트워크 이용률이 감소됨을 알 수 있다.

그림 10의 실험에서 x1의 네트워크 이용률은 평균 47.89%, m0의 네트워크 이용률은 50%로 측정되었다. 이번 실험 결과에서 두 호스트가 같은 개수의 소켓 연결을 이용하여 데이터를 전송할 경우, 네트워크의 대역폭은 1/2씩 사용함을 알 수 있다.

그림 7~10의 실험 결과를 종합해 보면, x1과 m0가 SC_{64}와 SC_2를 이용할 경우 네트워크 이용률은 평균 90.16%와 9.53%로 측정되었고, SC_{64}와 SC_{16}개의 경우에는 평균 76.27%와 23.82%, SC_{64}와 SC_{32}의 경우에는 평균 58.69%와 39.93%, SC_{64}와 SC_{64}의 경우에는 평균 47.89%와 50%의 네트워크 이용률이 측정되었다. 이러한 결과는 두 도메인 간에 사용 중인 소켓 연결의 총 개수 중에 현재 호스트에서 사용하는 소켓 연결 개수의 비율로 현재 호스트가 네트워크 대역폭을 이용하고 있음을 증명하였다. 이번 실험 결과로 네트워크 대역폭은 생성된 소켓 연결 간에 공평하게 나누어 사용되고 있음을 알 수 있다.

4. 네트워크 대역폭 할당 값 예측 모델링

그리드에서는 기관 간의 자원 공유를 위해 새로운 가상 조직이 구성되는데, 이를 VO (Virtual Organization)라 하며, PKI 기반의 상호 보안 협약



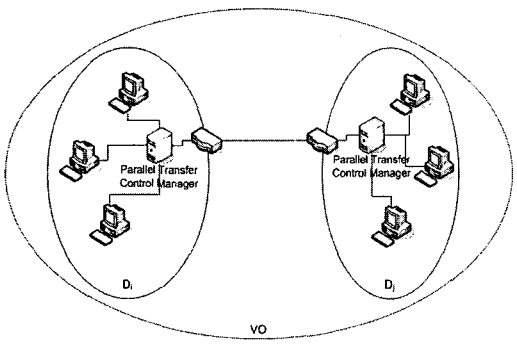
〈그림 10〉 두 호스트가 각각 64개와 64개 스트림을 이용하여 데이터 전송 시 간섭 현상

이 이루어짐으로써 기관간의 안전이 보장된다.

그림 11은 본 연구에서 설계한 VO 단위에서 병렬 데이터 전송 시 도메인 간의 연결 관리 시스템 구조를 나타내고 있다. VO 에는 도메인 i와 또 다른 도메인 j가 존재하고, 두 도메인 간에는 병렬 전송을 관리하는 Parallel Transfer Control Manager (PTCM)가 존재한다고 가정한다. 또한, 각 도메인 안에는 여러 호스트가 존재하며, 각 호스트에는 병렬 전송을 지원하는 응용 프로그램이 실행된다고 가정한다.

한 호스트가 다른 도메인의 한 호스트에 데이터를 전송하고자 할 때, 송신측 호스트는 데이터 전송에 필요한 대역폭을 PTCM에게 신청하고, PTCM 은 다른 소켓 연결들의 서비스 레벨을 고려하여, 요청한 대역폭 제공을 위한 연결 설정 정보(소켓 연결 개수 등)를 송신 호스트에게 알려준다. 만약 송신측 호스트가 제공받을 수 없는 대역폭을 요청했을 경우에는 PTCM은 서비스 제공 불가 메지시를 전송한다. 요청한 대역폭 사용이 가능한 경우에는 송신 호스트는 수신한 연결 설정 정보를 이용하여, 수신측과 연결을 설정하고 데이터를 전송한다. 데이터 전송이 완료되면, 설정된 연결의 종료를 PTCM에게 통보하고, PTCM은 병렬 전송에 관한 정보를 갱신한다.

하나의 소켓 연결이 네트워크 대역폭을'1/총 소켓 연결 수'만큼 할당받는다는 특징을 이용하여 새로 전송을 수행할 응용 프로그램이 요구하는 대역폭을 확보하기 위해 필요로 하는 소켓 연결의 개수를 계산할 수 있다.



〈그림 11〉 데이터 병렬 전송을 위한 연결 관리 시스템 구조

도메인 i(D_i)와 도메인 j(D_j)에서 현재 데이터 전송을 위해 실행중인 응용프로그램(t)의 소켓 연결 수를 At_{ij}라 하면, D_i와 D_j간에 전송되는 모든 소켓 연결 수의 합을 V_k라 하고, 다음 식 ①과 같이 표현할 수 있다.

$$V_k = \sum_{\forall t} A_{t_{ij}} \quad ①$$

이때, D_i 와 D_j 간에 새로운 연결을 통해 데이터를 전송하고자 하는 응용 프로그램(n)에서 새로 생성될 소켓 연결의 수를 An_{ij}라 하면, 새로 생성된 소켓 연결들이 할당받을 수 있는 대역폭(Bandwidth for New Allocation:BNA)은 식 ②와 같이 계산될 수 있다.

$$BNA = B \times \frac{A_{n_{ij}}}{V_k + A_{n_{ij}}} \quad ②$$

(B : D_i 에서 D_j 까지의 네트워크 대역폭)

새로 데이터 전송을 수행할 응용 프로그램이 요청한 대역폭을 획득하기 위해 필요한 연결의 수를 다음 식 ③과 같이 결정할 수 있다.

$$A_{n_{ij}} = \frac{V_k \times BNA}{B - BNA} \quad ③$$

그러나 새로운 연결을 생성하여 데이터 전송을 수행하고자 하면, 기존의 연결들은 새로운 연결에 의해 간섭을 받게 되므로, 기존 연결들이 이용하던 대역폭은 감소하게 된다. 이러한 상황에서 소켓 연결들의 서비스 레벨에 따른 대역폭을 유지하기 위해서는 새로운 연결의 생성된 이후 기존 연결들이 사용할 수 있는 대역폭을 예측할 수 있어야 한다.

새로운 소켓 연결들이 생성된 후 기존 연결들이 사용할 수 있는 대역폭(Interfered Bandwidth:IB)은 다음 식 ④로 예측할 수 있다.

$$IB = B \times \frac{V_k}{V_k + A_{n_{ij}}} \quad ④$$

그러므로 기존에 데이터를 전송하던 한 응용 프

로그램이 새로운 연결에 의해 간섭받은 이후 사용할 수 있는 대역폭은 식 ⑤로 예측할 수 있다.

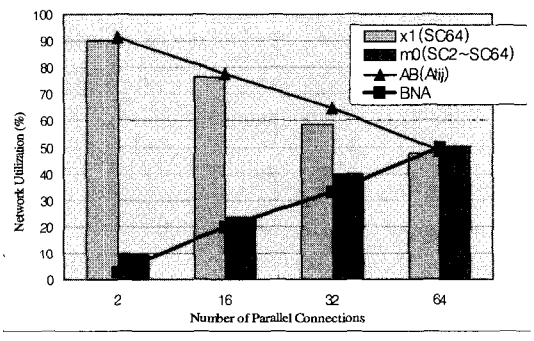
$$AB(A_{t_{ij}}) = IB \times \frac{A_{t_{ij}}}{V_k} \qquad ⑤$$

다음 그림 12는 위의 식 ②와 ④를 이용하여 x1과 m0의 소켓 연결이 실제 사용하는 대역폭과 예측값을 비교한 그래프이다. x1과 m0의 막대그래프는 그림 7~10의 각 실험에서 측정된 결과를 평균한 대역폭이며, 선형 그래프는 식 ②와 ⑤를 통해 예측된 대역폭이다. AB(At_{ij})는 데이터를 전송 중이던 응용 프로그램이 간섭 이후 사용하는 대역폭 예측값이고, BNA는 데이터 전송을 수행할 응용 프로그램이 사용하게 될 대역폭 예측값을 나타낸다.

그림 12에서 x1와 m0의 실제 값과 예측값인 BNA, AB(At_{ij})가 거의 일치하고 있음을 알 수 있다.

표 1은 3장의 실험에서 측정된 네트워크 활용률과 위의 식②와 ⑤를 이용하여 얻어진 예측값을 비교한 것이다. 3장의 실험은 m0의 응용프로그램이 데이터 전송 중에 x1의 응용프로그램이 새로운 연결을 설정하고 데이터를 전송하면서 수행되었기 때문에, m0에서 측정한 값과 NCB를 비교할 수 있으며, x1에서 측정된 값은 새로운 연결을 설정하여 획득할 수 있는 네트워크 대역폭이므로 AB(At_{ij})와 비교할 수 있다.

표 1을 보면 실험에서 얻어진 측정값과 예측값의 오차가 8%이하로 나타나 예측값의 정확성을 확인할 수 있다. 또한 위와 같은 결과는 PTCM이 호스트 간의 소켓 연결의 수를 통하여 대역폭의 제어가 가능함을 의미하며, 이는 QoS 지원이 가능함을 의미한다.



〈그림 12〉 실제 사용 대역폭과 예측 대역폭의 비교

〈표 1〉 실제 측정값과 예측값의 오차 비교 (%)

	X1(SC_{64})	AB(At_{ij})	Error	m0(SC_2~ SC_{64})	NCB	Error
SC_2	90.16	96.97	6.81	9.53	3.03	6.50
SC_{16}	76.27	80.00	3.73	23.82	20.00	3.82
SC_{32}	58.69	66.67	7.98	39.93	33.33	6.60
SC_{64}	47.89	50.00	2.11	50.06	50.00	0.06

5. 결론 및 향후 연구 과제

본 논문에서는 네트워크 성능 향상을 위하여 연구되고 있는 DRS, XCP등의 기술들을 살펴보았다. 그 중, 병렬 전송 기술에 대해 논하고, 여러 실험을 통하여 병렬 전송 시 소켓 연결들 간에 간섭 현상이 발생함을 실험을 통하여 확인하였다.

따라서 병렬 전송 기술을 효율적으로 활용하기 위해서는 병렬 전송을 관리할 수 있는 메커니즘이 요구된다 할 수 있는데, 이 메커니즘 실현을 위해서는 새로운 연결로 인해 기존의 연결들이 어느 정도 영향을 받는지를 예측할 수 있어야 하며, 새로운 데이터 전송 시 요구되는 대역폭 획득을 위해 몇 개의 소켓 연결이 필요한지를 계산할 수 있어야 한다.

따라서 본 논문에서는 실험을 통하여 증명한 병렬 전송 특징(한 소켓 연결이 네트워크 대역폭을 '1/총 소켓 연결 수'만큼 할당받는다)을 이용하여 새로운 연결의 개수 계산과 기존 응용 프로그램의 대역폭 예측 값을 모델링하였다.

앞으로는 본 논문에서 제시한 모델링을 통하여 QoS가 지원되는 데이터 병렬 전송 관리 메커니즘 연구가 이루어져야 할 것이다.

참 고 문 헌

[1] The Data Grid: Towards an Architecture for the Distributed Management. and Analysis of Large Scientic Datasets. Ann Chervenak, Ian

Foster, C.Kesselman, C.Salisbury, S.Tuecke, Journal of Network and Computer Application, 23, 187-200, 2001

[2] GridFTP v2 Protocol Description, Bill Allcock, Igor Mandrichenko, Timur Perelmutov, Germi National Accelerator Laboratory, 2004

[3] GridFTP: Protocol Extensions to FTP for the Grid, W. Allcock, Argonne National Lab.

[4] GridFTPexplained, Pawel Plaszczak, Joe Link, Rich Wellner, Paul Hubbard, www.icslab.agh.edu.pl/~kzajac/pawel/GridFTPexplained.doc

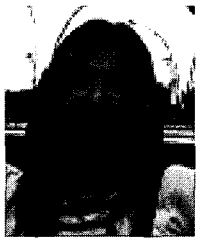
[5] J.Semke, J.Mahdavi and M.Mathis, "Automatic TCP Buffer Tuning", ACM SIGCOMM 1998, vol. 28, no.4, 1998

[6] Eric Weigle and Wu-chun Feng, "Dynamic Right-Sizing:A Simulation Study." IEEE ICCCN, 2001.

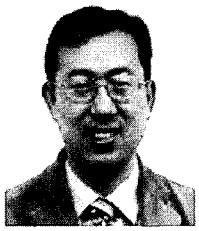
[7] Aaron falk, Ted Faber, Joseph Bannister, Andrew Chien, Robert Grossman, Jason Leigh, "Transport Protocols for High Performance", Communications of The ACM, vol. 46, no 11, November 2003.

[8] An Efficient TCP Buffer Tuning Technique Based on Packet Loss Ratio (TBT-PLR), Gi-chul Yoo, Eun-sook Sim, Dongkyun Kim, Taeyoung Byun, Kookhan Kim, Okh- wan Byun, International Conference on In- ternet Computing, 2004

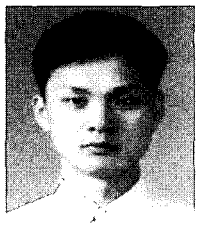
◐ 저 자 소 개 ◑



김 영 신(Young-Shin Kim)
1999년 서울여자대학교 컴퓨터학과 졸업(학사)
2002년 서울여자대학교 대학원 컴퓨터학과 졸업(석사)
2004년 서울여자대학교 대학원 컴퓨터학과 수료(박사)
2002년~현재 서울여자대학교 대학원 박사과정
관심분야 : 데이터 그리드, 텔레메틱스
E-mail : amary46@swu.ac.kr



허 의 남(Eui-Nam Huh)
1985년 부산대학교 전산통계 (학사)
1995년 University of Texas, 전산학 (석사)
2000년 The Ohio University, 전산공학 (박사)
2002년 삼육대학교
2003년 서울여자대학교 컴퓨터공학과 조교수
2005년~현재 경희대학교 컴퓨터공학과 조교수
관심분야 : Telemetics, Grid, Ubiquitous, embedded system
E-mail : johnhuh@khu.ac.kr



김 일 중(Il-Jung Kim)
1998년 전남대학교 산업공학과 졸업(학사)
2001년~2002년 LG전자 차세대통신연구소 주임연구원
2002년~2005년 KTF 테크놀러지스 차세대단말연구소 선임연구원
2005년~현재 한국전산원 차세대인터넷팀 근무
관심분야 :
E-mail : ijkim@nca.or.kr

황 준 (Jun Hwang)
1985년 중앙대학교 컴퓨터공학과 졸업(학사)
1987년 중앙대학교 대학원 컴퓨터공학과 졸업(석사)
1991년 중앙대학교 대학원 컴퓨터공학과 졸업(박사)
1992년~현재 서울여자대학교 정보통신공학부 교수
관심분야 : 유비쿼터스, 분산 처리.
E-mail : hjun@swu.ac.kr