데이타 그리드상에서 TCP 버퍼의 PID 제어를 통한 QoS 구현

(QoS Implementation by using PID Control of TCP Buffer in Data Grid)

이 홍 석 † 허 의 남 †† 추 현 승 ††† 조 금 원 ††††
(Hongseok Lee) (Eui-Nam Huh) (Hyunseung Choo) (Kum Won Cho)

요 약 데이타 그리드 상의 빈번한 대용량의 파일 전송으로 급격히 증가하는 트래픽에 따른 대역폭 관리는 이제 네트워크 운영에서 필수적인 요소로 인식되고 있다. 본 논문에서는 TCP 버퍼 튜닝 연구를 통해 대역폭을 조정하는 기존의 연구를 바탕으로 TCP 계층의 자원을 제어함으로써 QoS를 보장하는 새로운 기법을 제안한다. 일반적으로 QoS솔루션은 IP 계층이나 링크 계층에서 네트워크 자원을 관찰하고 관리하지만, 본 논문에서는 네트워크 상위 계층인 TCP 계층에서 네트워크 자원을 제어한다. 그 결과 각 사용자에게 부여된 권한에 따라 서로 다른 대역폭을 사용할 수 있도록 효율적으로 조절할 수 있다. 이는 네트워크 자원관리와 과금에 새로운 패러다임을 제공할 것으로 기대한다.

키워드 : 데이타 그리드, TCP 버퍼 튜닝, QoS 구현, PID제어, 네트워크 자원관리

Abstract It is aware that Bandwidth management with dramatically increasing traffic on account of frequent and large file transmission in a data grid environment is one of essential needs. A this paper we propose new method which guarantees QoS (Quality of Service) by being in control of resources in TCP layer based on existing studies that manage bandwidth over TCP buffer tuning. General QoS solutions manage network resources subsequent to observing them in IP or link layer, but the scheme in the paper is able to control network resources in TCP layer that is network upper layer. Consequently, bandwidth allocation to each user can be efficiently controlled depending on an authority each user is given so that users could be use different bandwidth. It is expected that a new paradigm is supposed in network resource management and the method of levies for users' bandwidth uses.

Key words : Data Grid, TCP Buffer Tuning, QoS Implementation, PID Control, Network Resource Management

1. 서 론

계속적으로 복잡해지는 웹 어플리케이션과 급격히 증가하는 트래픽으로 인해 대역폭 관리는 이제 네트워크 운영에서 필수적인 요소로 인식되고 있다. 최근 대부분의 네트워크에서, 급격히 증가하고 있는 웹 관련 트래픽이 중요도에 상관 없이 많은 대역폭을 사용하여 정작 신속하게 처리되어야 할 중요한 서비스의 처리를 방해하고 있는 경우가 많다. 데이타 그리드 상에서는 많은 사용자가 대용량의 파일을 전송하는 경우가 매우 많다. 이 때 트래픽 또는 사용자에 따라 우선 순위를 부여하고 중요도에 따라 최소, 최대 대역폭을 설정하는 방법이 필요하다. 이를 통하여 중요한 업무 관련 트래픽은 최소 대역폭 이상이 되도록 규정하고 업무와 무관한 웹 관련 트래픽은 최대 대역폭을 설정하여 대역폭 이상은 사용할 수 없도록 제한하여 효과적인 네트워크 환경을 보장하게 한다[1]. 이를 위해서는 QoS(Quality of Service)를 지원해야 한다. QoS란 사용자 또는 어플리케이션에 대해 중요도에 따라 서비스 수준을 차등화하여 한정된

· 본 연구는 정보통신부 및 정보통신연구진흥원의 대학 IT연구센터 지원 사업의 연구결과로 수행되었음. [ITA-2005-(C1090-0501-0019)
† 비 회 원 : 성균관대학교 정보통신공학부
juspeace@hanmail.net
†† 정 회 원 : 경희대학교 전자정보학부 교수
johnhuh@khu.ac.kr
††† 종신회원 : 성균관대학교 정보통신공학부 교수
choo@ece.skku.ac.kr
†††† 정 회 원 : 한국과학기술정보연구원 슈퍼컴퓨팅 응용지원 팀장
ckw@kisti.re.kr
논문접수 : 2006년 3월 24일
심사완료 : 2006년 4월 7일

대역폭에서 트래픽과 대역폭을 정책적으로 관리하는 기술을 말한다. QoS 솔루션은 대역폭과 그 안에서 발생하는 트래픽을 모니터링과 분석을 통해 효과적으로 제어하고 관리할 수 있는 기술을 제공한다. 이를 데이타 그리드에 적용하면 각 사용자에 권한을 조정하는데 매우 효율적이다[2].

데이타 전송에 기반이 되는 TCP(Transmission Control Protocol)는 신뢰성 있는 통신을 이루기 위해 널리 사용되는 인터넷 프로토콜[3]이다. 하지만 불행하게도 TCP는 고성능 네트워크나 고성능 컴퓨팅 환경을 염두하고 설계된 것이 아니라 대역폭을 공평하게 나누어 쓰고자 하는 목적으로 형평성에 중점을 두고 설계되었기 때문에 사용자는 TCP 성능을 최적화 하는 작업을 수동으로 수행해야 한다. 가장 중요하고도 어려운 작업이 적절한 TCP 버퍼 크기를 결정하고 설정하는 것이다. 하지만 이러한 작업들이 매번 수동으로 이루어질 수 없기 때문에 자동으로 높은 처리량을 얻기 위한 TCP 버퍼 튜닝 연구가 세계적으로 활발하게 진행 중이다[4,5]. 기존에 TCP 버퍼 튜닝과 관련이 있는 연구들 중에서 유명한 것으로는 DRS(Dynamic Right Sizing)[6], ATBT(Automatic TCP Buffer Tun-ing)[7], Linux auto-tuning[8] 등이 있다. 이들 기법들은 주로 기존의 TCP에서 버퍼를 얼마만큼 할당하여야 메모리를 낭비 안하고 TCP의 성능을 최대한으로 끌어 올릴 수 있는지에 대한 답을 제시한다. 보통 TCP 버퍼 크기가 크면 TCP 성능은 좋지만 메모리를 낭비하게 되고, 버퍼의 크기가 작으면 메모리는 아낄 수 있지만 TCP 성능이 떨어지기 때문에, 이를 적절히 조절하는 메커니즘을 제시하는 것이 TCP 고속전송 연구에서 주된 주제가 된다[9].

본 논문에서는 TCP 버퍼 튜닝 연구의 결과를 활용하여 TCP 계층의 자원을 제어함으로써 QoS를 구현하는 새로운 연구를 시도한다. 일반적으로 QoS솔루션은 IP 계층이나 링크 계층에서 네트워크 자원을 관찰하고 관리하지만, 네트워크 상위 계층인 TCP 계층에서 네트워크 자원을 제어하고자 한다. QoS를 구현하기 위해서는 우선순위를 부여하기 위해 응용프로그램의 종류나 사용자를 구분해야 하는데, 네트워크 하위 계층에서 이를 수행하는 일은 적지 않은 오버헤드를 요하는 일이다. 왜냐하면 네트워크 하위 계층에서 이를 수행하기 위해서는 상위 계층에서 만들어진 패킷을 다시 분석하는 일이 필요하기 때문이다. 하지만 이러한 작업을 네트워크 상위 계층에서 수행하게 되면 좀더 작은 오버헤드로 같은 일을 수행할 수 있기 때문에 더 효율적인 QoS를 수행할 수 있다. 물론, TCP 계층에서 제어할 수 없는 부분은 하위 IP 계층이나 링크 계층에서 이루어져야 한다. 그렇기 때문에 네트워크의 모든 계층이 QoS를 수행하는 것이 최상의 솔루션이다. 하지만 지금까지 TCP 계층에서 QoS를 보장하기 위한 연구는 이루어 지지 않고 있으며, 본 논문을 통해 TCP 계층 QoS 연구의 기반을 제공하고자 한다.

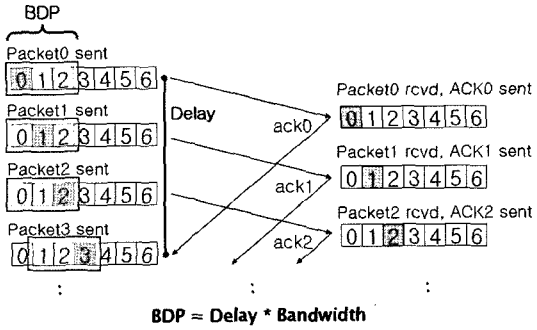

그림 1 BDP의 개념

논문의 나머지 부분은 다음과 같이 구성하였다. TCP 버퍼, PID 제어기, 그리고 일반적인 QoS 메커니즘에 관련한 연구내용을 2절에서 정리하며, 3절에서는 QoS를 TCP상에서 구현하기 위한 방법 소개와 구현된 프로그램에 대해서 기술한다. 4절에서는 TCP 버퍼를 제어하여 QoS를 수행하는 시스템을 구성하고 성능을 측정한 후 그 결과를 분석한다. 마지막으로 5절에서 결론을 제시한다.

2. 관련 연구

2.1 TCP 버퍼

TCP 버퍼란 송신측과 수신측에서, 통신을 수행하기 위해 물리적으로 할당된 메모리 영역을 말한다. 송신측에서의 TCP 버퍼는 한 번에 보낼 수 있는 물리적인 한계를 의미하며, 수신측에서의 TCP 버퍼는 수신측의 수용 능력을 의미한다. 수신측의 TCP 버퍼는 흐름제어를 통해 송신측에 그 정보가 전달되기 때문에 송신측은 그 정보에 따라 전송 속도를 조절한다. 결국 TCP 버퍼의 크기는 전송 속도와 직접적인 관련이 있다. 물리적인 한계가 전송 속도를 제한할 수 있다는 것을 뜻하기 때문에 시스템의 TCP 버퍼 설정 값이 충분히 크지 않다면 뜻하지 않은 전송 속도 제한을 경험하게 된다. 그렇다고 TCP 수신버퍼의 크기를 무한정 크게 설정할 수는 없다. TCP 버퍼의 크기가 작으면 전송 속도를 제한하게 되지만, 너무 크면 메모리를 낭비하게 되기 때문이다. 따라서 이를 적절한 값으로 결정할 수 있는 기준이 있어야 하는데, 기본적인 기준으로서 BDP(Bandwidth Delay Product)가 있다[10].

여기서 Delay란 TCP에서 송신측이 패킷을 보내고

그 패킷에 대한 ACK를 받을 때까지의 시간을 말하며 Bandwidth란 통신의 대역폭이 결정되는 중간 병목구간의 대역폭을 말한다. 그림 1에서 BDP의 개념을 함축적으로 설명하고 있다. 이론적으로 BDP의 값이 TCP 버퍼의 가장 적당한 크기이다. 이와 같이 TCP 버퍼의 가장 적절한 크기를 구해서 TCP 버퍼 크기를 설정하는 행위를 TCP 버퍼 튜닝이라고 한다. BDP의 값이 TCP 버퍼를 튜닝 함에 있어서 작지도 않고 크지도 않은 적당한 크기를 결정하는 좋은 기준이 된다. 일반적으로 TCP 튜닝은 큰 BDP를 가지는 네트워크 환경에서 작은 TCP 버퍼를 사용하고 있을 때 TCP 버퍼 크기를 BDP에 맞게끔 키워주는 행위를 말한다. 하지만 이론적인 BDP 값이 실제로 TCP 버퍼 크기의 가장 적절한 값인지는 실험을 통한 검증이 필요하다.

[11]논문은 BDP와 관련된 실험을 한 논문으로 수원 S 대학교와 부산 P 대학교 간의 대역폭 측정 실험에서는 그림 2와 같은 결과를 얻었는데, BDP수원-부산는 '20msec * 80Mbps'로 대략 200KByte이기 때문에 TCP 버퍼의 크기가 이 크기보다 작은 구간에서는 TCP 버퍼가 제한 요소가 되어 대역폭을 제한하지만 이 크기보다 큰 구간이 되면 TCP 버퍼는 더 이상 제한 요소가 되지 않는다. 그래프에서 TCP 버퍼 크기가 200KByte가 될 때까지는 대역폭이 증가하다가 200KByte이후로 대역폭이 일정하게 유지되는 것은 바로 이러한 이유에서다.

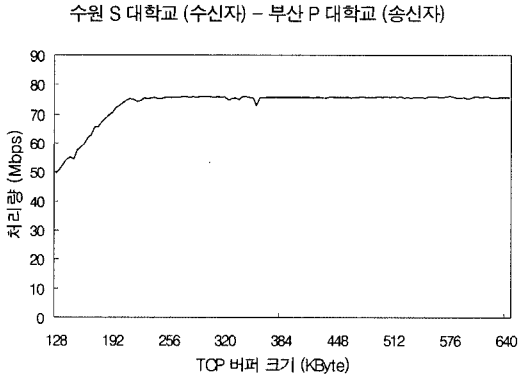

그림 2 TCP 버퍼 크기에 따른 처리량 변화

2.2 PID 제어기

PID(Proportional-plus-Integrate-plus-Derivative)는 어떤 제어 대상에 대해서 정교한 조절 능력을 얻을 수 있도록 해주는 메커니즘이다. 제어 과정에서 실제 측정값이 꾸준하게 제어기에게 피드백 되며, 희망하는 값이 대상 시스템에서 실현될 수 있도록 꾸준히 노력한다. 제어기의 임무는 대상 시스템의 측정값을 목표 값과 같아지도록 하는 것이다. 전문용어로, 측정값을 process value라 하며 목표 값을 set point라 한다. 이러한 작업을 달성하는 가장 좋은 방법 중에 하나가 PID 제어기로 알려진 제어 알고리즘을 사용하는 것이다. PID 제어기는 3가지 수학적인 제어 기능을 포함하며 이들은 서로 맞물려 동작한다. 그 3가지는 Proportional, Integral, Derivative이다. 이 중에 가장 중요한 Proportional 제어기는, 에러 값이라 할 수 있는 process variable과 set point의 차이 값을 결정하여 이 에러를 제거하기 위해 제어 변수에 적절히 변화를 준다. 실제로, 대부분의 제어 시스템은 오로지 이 Proportional 제어기만을 사용하여도 아주 잘 동작한다. Integral 제어기는 시간이 지남에 따라 축적된 set point과 process variable의 크기를 유지함으로써 필요할 경우 이를 활용하여 대상 시스템을 제어한다. Derivative 제어기는 process variable의 변화율을 꾸준히 관찰하여 제어기가 비정상적인 변화에 대해 적절히 대처할 수 있도록 해준다. PID 제어기를 이용하면 네트워크 통신과 같이 실험환경이 그때그때 변화하는 경우에도 process variable을 원하는 수치 즉, set point에 상당히 근접하게 접근시킬 수 있다는 장점이 있다[12].

P 기능, I 기능, 그리고 D 기능, 이들 각각의 3가지 기능들이 수행되기 위해서는 PID 인자라고 불리 우는 값들을 사용자 스스로 결정해야 한다. 이 인자 값들은 제어 시스템마다 너무나도 다양해서 제어의 정확성을 최적으로 끌어올리려면 인자를 결정하기 위한 적절한 조정 작업이 필요하다. 본 논문에서는 일반적인 시스템에서 최적이라고 알려져 있는 PID 인자 값을 사용하였고, 실험을 통해 PID 인자 값을 수동으로 적절히 조절하였다. 하지만 이러한 인자 값들을 자동으로 결정하는 과정이 있으며, 이를 PID 튜닝이라고 한다. 본 논문에서는 TCP 버퍼의 최적 크기를 결정하기 위해 PID 제어기를 사용하지만 PID 튜닝은 수행하지 않는다. 적절하게 PID를 튜닝을 수행하는 작업은 향후 연구 주제로 남겨놓는다.

2.3 QoS 매커니즘

QoS 기술은 크게 QoS 보장 기술과 제공된 QoS의 상태를 측정하기 위한 QoS 모니터링 기술로 나뉘어 진다. QoS 보장 기술은 다시 각 네트워크 장비에서 제공되어야 될 트래픽 관리 기술, 네트워크 전체 입장에서 QoS의 보장 기술 및 이를 관리할 수 있는 정책기반의 QoS 관리 기술로 나뉘어 진다. 또한 QoS 모니터링 기술은 프로토콜 모니터링, 네트워크 모니터링, 종단간 QoS 모니터링 기술로 세분된다[13]. 본 논문에서 구현하고자 하는 TCP 버퍼 제어에 의한 QoS 구현은 QoS 보장 기술에 속하며, 그 중에서도 트래픽 관리 기술에 속한다. 트래픽 관리 기술은 다양한 세부 기술들이 소개

되어 있으며 크게 큐관리, 트래픽 쉐이핑, 수락제어, 폴리싱, 혼잡관리의 분야로 나누어진다[14]. 이들 각각은 독립적으로 사용될 수도 있지만 대표적인 QoS 관리 모델로 알려진 통합서비스 모델과 차등서비스 모델에서와 같이 복합적으로 사용된다.

3. 제안하는 방식

3.1 QoS 보장을 위한 TCP 버퍼 튜닝

기존에 개발된 QoS 솔루션들은 네트워크 장비 단에서 우선순위를 두어 패킷 단위로 QoS를 수행하는 방식이 주를 이룬다. 이러한 QoS 기술들은 실제로 사용자에게 어느 정도 차별화된 서비스를 제공해 줄 수는 있지만, 사용자에게 적정한 대역폭을 할당하고 해당 대역폭의 보장하는 서비스를 제공하는 데에는 어려움이 있다. 왜냐하면 IP 계층의 동작방식 자체가 응용계층의 처리량과 직접적인 연관이 없기 때문이다. 단지 간접적으로 일부 응용프로그램의 패킷을 우선적으로 처리해 줄 수 있을 뿐이다. 하지만 TCP 계층의 TCP 버퍼는 응용계층의 처리량과 매우 밀접한 관련이 있다. 그림 3은 TCP버퍼 크기와 처리량의 관계를 보여준다. TCP 버퍼 크기에 따른 처리량의 변화가 비례 관계임을 보여주고 있다. 본 논문에서는 이 점을 이용하여 사용자에게 필요한 대역폭을 설정하고 이와 비교하여 현재 전송되고 있는 데이타량을 조정하기 위해서는 TCP 버퍼 크기와 전송 대역폭이 선형적 비례관계라는 선행 연구결과를 사용한다.

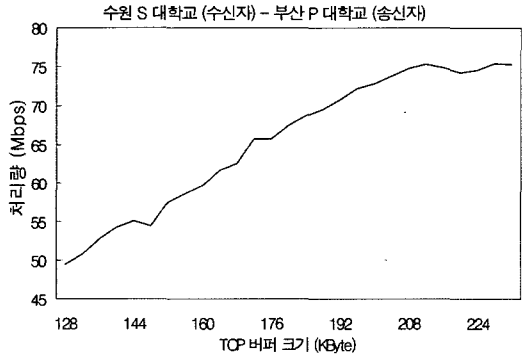

그림 3 TCP버퍼크기와 처리량의 비례관계

이와 같이 처음 결정한 대역폭값과 실질적으로 응용프로그램의 데이타 전송량을 비교하여 차이가 있을 경우 이 때 자동적으로 TCP 버퍼 사이즈를 조정하는 장치가 필요하다. 이를 위해서 본 연구에서는 PID제어기를 사용하여 전송량의 흐름 제어를 구현한다.

3.2 PID 제어기를 사용한 흐름 제어

또한 본 프로그램에서는 PID 제어기를 사용하는데, PID 제어기를 사용하기 위해서는 PID 인자를 적절하게 결정하는 작업이 필요하다. 이를 위해서 본 논문에는 다음과 같은 수식의 PID 제어기 엔진을 사용하였다.

Symbol Definition
* impact : 조정할 TCP 버퍼 사이즈
* Pk : PID 제어기의 P부분 파라미터
* Dk : PID 제어기의 D부분 파라미터
* need_val : 필요한 전송 대역폭
* old_val : 이전의 필요 전송 대역폭

```
impact = ((Pk * need_val) +
          (Dk * (need_val - old_val)))
```

PID 인자는 시스템의 특성에 따라 다양한 값을 가지고 본 논문에서는 P인자를 1로 설정하여 현 대역폭을 얻기위해 필요한 버퍼 사이즈를 바로 도달하기도록 설정하고, 이를 보정하기 위해서 이전 대역폭과의 차이에 10%만큼 더 버퍼사이즈를 할당하도록 D인자를 0.1로 설정한다. PID제어기의 I인자는 사용하지 않으므로 0으로 설정한다. PID제어기의 각각의 인자는 시스템마다 유동적일 수 있으며 경우에 따라서 자동으로 PID 인자값을 결정할 수 있다. 이를 PID 튜닝이라고 하며 본 연구에서는 PID를 수동으로 결정하였으며 PID 튜닝 문제는 향후 연구 주제로 남겨놓는다.

표 1 성능평가에서 사용한 PID 인자값

	정 의	값
P	Coefficient of the proportional part	1.0
I	Coefficient of the integral part	0
D	Coefficient of the derivative part	0.1

3.3 QoS 보장을 위한 시스템 구현

본 논문에서는 TCP 계층을 제어함으로써 QoS를 구현하고자 하는 새로운 시도를 한다. TCP 계층을 제어한다는 말은 TCP 버퍼를 제어하는 것을 의미하며, QoS를 구현한다는 말은 사용자가 특정 대역폭을 요청하면 처리량이 해당 대역폭에 수렴할 수 있도록 해준다는 것을 의미한다. 즉, TCP 계층의 TCP 버퍼를 제어함으로써 원하는 대역폭을 얻을 수 있도록 하는 프로그램을 개발한 것이 본 연구에서 수행한 내용이다. 본 시스템은 응용계층의 라이브러리 형태로 구현되어 있다[8]. 소켓 통신을 사용하고자 하는 응용프로그램이 본 시스템의 함수를 사용하여 통신을 수행하면 내부적으로 QoS 알고리즘이 동작하도록 구현되어 있다. 그림 4는

QoS 보장 시스템의 전체 구조이다. 사용자가 수동으로 필요한 대역폭을 설정하면 시스템은 QoS 초기화 모듈에서 필요 대역폭을 기반으로 각 변수를 초기화한다. 이를 기반하여 QoS 전송 모듈이 리눅스 커널의 소켓전송을 사용하여 데이타를 보내며 이 때 대역폭을 조정하기 위해서 PID 제어기와 연동하여 동작한다.

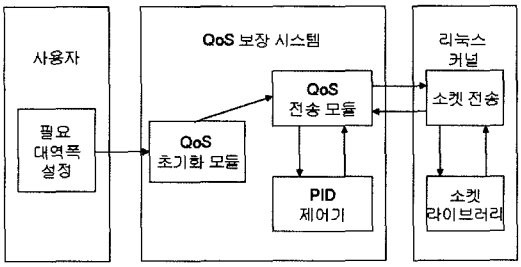

그림 4 QoS 보장 시스템 전체 구조

• QoS 초기화 모듈

응용프로그램에서 원하는 대역폭을 설정할 때에 사용하는 함수이다. TCP 소켓을 생성한 이후에 동작하며, 실질적인 데이타 송신이 이루어지기 이전에 QoS 전송 기본값을 초기화한다. 여기서 설정된 초기값을 사용하여 QoS 전송모듈이 동작한다.

• QoS 전송모듈

응용프로그램이 TCP 소켓을 이용하여 데이타를 송신하고자 할 때에 사용하는 모듈이다. 기존의 소켓프로그래밍에서 데이타를 보내고자 할 때 사용하는 send()라는 함수와 연동하여 동작한다. 또한 QoS 전송을 하기 위하여 5초마다, 응용프로그램이 전송하는 처리량을 계산해서 사용자가 요구하는 대역폭에 도달하는 지를 검사한다. 만약 해당 대역폭에 도달하지 않아서 처리량을 변화해야 하는 상황이 발생하면 PID 제어기를 호출한다.

• PID 제어기

사용자가 요구하는 대역폭과 실제 응용프로그램이 사용하는 처리량의 차이를 바탕으로 TCP 버퍼 크기를 결정하고 설정하는 역할을 한다. PID 제어기에서 결정된 TCP 버퍼 크기는 결국 응용프로그램의 처리량을 변화시키며 사용자가 요구하는 대역폭에 좀더 가까이 도달하도록 해 준다.

4. 성능 평가

4.1 실험 환경 및 과정

TCP의 버퍼를 튜닝 하기 위해 개발된 프로그램의 성능평가를 위해서 다음과 같은 환경을 구축한다. 데이타를 주고받을 두 개의 호스트 중에서 송신자는 개발된 QoS 라이브러리를 이용하는 소켓프로그램이 수행하도록 코드를 실행하고 수신자는 일반적인 TCP 소켓프로그램이 실행하도록 하여 환경을 구축한다. 그림 5는 현재 데이타 전송 실험환경을 표현한다.

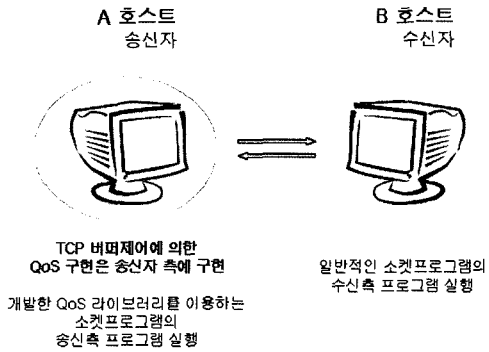

그림 5 QoS 구현을 위한 데이타 전송 실험환경

KOREN 기가 대역폭 망을 사용하고 0.5 msec 지연시간을 가지는 거리에 위치한 두 호스트가 본 프로그램의 성능평가 실험에 사용되었다. 실험에 사용된 두 개의 호스트의 자세한 시스템 명세사항은 표 1과 같다.

표 2 성능평가에 사용된 시스템의 명세사항

	A 호스트	B 호스트
OS	Linux kernel 2.4.20	Linux kernel 2.4.20
CPU	Pentium IV 3.0 GHz	Xeon 2.4 GHz (Dual)
RAM	771744 KB	1048576 KB
LAN Card	1 Gbps Ethernet	1 Gbps Ethernet

TCP 계층의 버퍼를 제어함으로써 QoS를 구현한 본 프로그램이 제대로 동작하는 지를 검증하기 위해 다음과 실험을 수행하였다. 송신을 담당하는 프로그램에 본 연구에서 개발한 QoS 라이브러리가 포함되도록 컴파일한다. 그리고 프로그램 내에서 대역폭 설정값을 100 Mbps로 설정한 후에 20분 동안 끊임없는 데이타 전송을 수행하도록 한다. 20분이 경과하면 대역폭 설정값을 200Mbps로 바꾸어 설정한 후에 다시 20분 동안 끊임없는 데이타 전송을 수행하도록 한다. 같은 방법으로 100 Mbps씩 요구 대역폭을 올리면서 1000Mbps까지 실험한다. 20분 동안 데이타를 전송시키는 동안 실제 처리량이 얼만큼 나오는 지를 기록하여 다음절에 결과를 보일 것이다. 프로그램 구현에서 언급했다시피 PID 제어는 5초에 한번씩 발생하기 때문에 20분 동안 상당히 많은 TCP 버퍼 조정작업을 하게 된다.

4.2 결과 분석

그림 6은 사용자가 요구 대역폭을 100Mbps에서 900 Mbps까지 100Mbps 간격으로 설정했을 때, 실제 통신을 수행하는 응용프로그램이 내는 처리량을 5초 간격으로 측정하여 막대그래프로 나타낸 그래프이다. 각 실험마다 총 20분 동안 실험을 수행했기 때문에 막대그래프가 촘촘하게 표현되어 있다. 그래프를 보면 알 수 있다시피 본 연구에서 개발한 QoS에 의해서 처리량이 요청한 대역폭에 거의 근접해 있다. 해당 대역폭을 경계로 약간 들쑥날쑥한 현상을 보이는데, 이는 PID 제어기가 해당 대역폭 주위에서 처리량을 예민하게 높이고 줄이는 일을 반복하고 있기 때문이다. 본 성능평가의 기반이 1Gbps 망에서 이루어지고 있기는 하지만 실제로 얻을 수 있는 최대대역폭은 대략 770Mbps 정도 밖에 되지 않기 때문에 800Mbps이상의 대역폭을 요청한 실험에서는 약 770Mbps 정도에서 안정화가 되는 모습을 관찰할 수 있다.

5. 결 론

본 연구에서는 TCP 계층의 TCP 버퍼 크기를 제어함으로써 사용자가 요청하는 대역폭을 수렴할 수 있도록 해주는 QoS 보장 시스템을 개발하였다. 데이타 그리드상에서 대용량의 데이타가 전송되며 또한 많은 사용자가 접속하여 이용된다. 이때 각 사용자마다 차별성을 주는 일은 반드시 필요한데 본 연구의 결과를 사용하면 한정된 대역폭을 꼭 필요한 사람에게 우선적으로 할당시킬 수 있는 환경이 효율적이고 손쉽게 제공될 수 있다. 또한 지금까지 TCP 계층에서 QoS를 보장하기 위한 연구는 이루어 지지 않고 있으나, 본 논문을 통해 TCP 계층 QoS 연구의 기반을 제공하였다. TCP 송신 버퍼를 적절하게 제어하기 위해 PID 제어 메커니즘을 도입하였고 PID 인자 값을 적절한 값으로 설정하여 나아진 성능을 확인하였다. 그렇다 할지라도 PID 튜닝이라는 과정을 거쳐서 PID 인자 값을 결정하면 더욱더 최

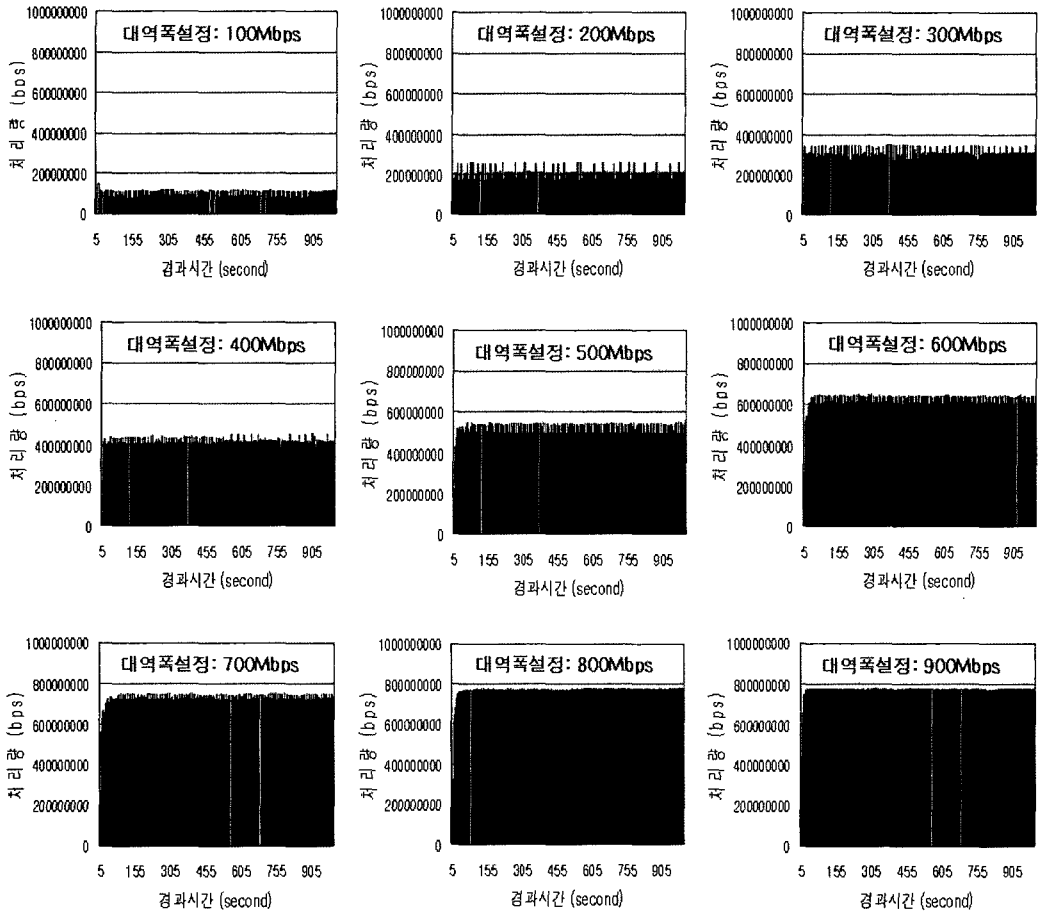

그림 6 TCP 버퍼 제어에 의한 대역폭 수렴

적화된 결과를 얻을 수 있다. 향후 연구로서, PID 튜닝을 거쳐 더욱 향상된 TCP QoS 메커니즘을 개발할 예정이다. 또한 데이타 그리드상에서 본 논문에서 구현된 시스템이 효율적으로 작동하기 위해서 다중 사용자에서의 QoS보장을 위한 연구도 중요한 과제이다.

참 고 문 헌

[1] D. Kandlur, D. Saha, and K. Shin, "Understanding TCP Dynamics in an Integrated Services Internet," NOSSDAV '97, May 1997.

[2] Paul Ferguson, and Geoff Huston, Quality of Service: Delivering QoS on the Internet and in Corporate Networks, John Wiley & Sons, Inc. 1998.

[3] W. Stevens, "RFC 2001: TCP Slow Start, Congestion Avoidance, Fast Retransmit, and Fast Recovery Algorithms," January 1997.

[4] J. Semke, J. Mahdavi and M. Mathis, "Automatic TCP buffer tuning," in Proceedings of ACM SIGCOMM '98, pp. 315-323, August 1998.

[5] Takahiro Mautsuo, Go Hasegawa and Masayuki Murata, "Scalable Automatic Buffer Tuning to Provide High Performance and Fair Service for TCP Connection," in proceedings of IEEE INET 2000, July 2000.

[6] E. Weigle and W. Feng, "Dynamic right-sizing: A simulation study," in Proceedings of IEEE ICCCN, October 2001.

[7] J. Semke, J. Mahdavi and M. Mathis, "Automatic TCP buffer tuning," in Proceedings of ACM SIGCOMM '98, pp. 315-323, August 1998.

[8] Linus Torvalds and The Free Software Community, "The Linux Kernel," September 1991, http://www.kernel.org/.

[9] Pittsburgh Supercomputing Center, "Enabling High-Performance Data Transfers on Hosts," http://www.psc.edu/networking/perf tune.html.

[10] Mike Fisk and Wu-chun Feng, "Dynamic Adjustment of TCP Window Sizes," http://public.lanl.gov/mfisk/fisk/web/papers/tcpwindow.pdf, July 2000.

[11] 박병철, 허의남, 조일권, 추현승, "TCP 버퍼 튜닝의 이론적 분석과 버퍼 크기에 따른 대역폭 변화 연구", 한국인터넷정보학회 추계학술대회 논문집, 제5권 2호, pp. 375-378, 2004년 11월.

[12] Andr´as Veres and Mikl´os Boda, "The Chaotic Nature of TCP Congestion Control," in Proceedings of IEEE Infocom 2000, March 2000.

[13] R. Braden, D. Clark, S. Shenker, "Integrated Services in the Internet Architecture: An Overview," RFC1633, June 1994.

[14] Paul Ferguson, and Geoff Huston, Quality of Service: Delivering QoS on the Internet and in Corporate Networks, John Wiley & Sons, Inc. 1998.

이 홍 석

1998년 성균관대학교 수학과 졸업(이학사). 2002년 성균관대학교 전기전자 및 컴퓨터공학부 졸업(공학석사). 2002년~현재 성균관대학교 정보통신공학부 박사과정. 관심분야는 이동컴퓨팅, 유무선 네트워크, 그리드 컴퓨팅

허 의 남

1990년 부산대학교 전산통계 학사. 1995년 텍사스대학교 전산학과 석사. 2002년 오하이오대학교 전산학과 박사. 2003년~2005년 8월 서울여자대학교 정보통신대학 조교수. 2005년 9월~현재 경희대학교 전자정보대학 컴퓨터공학과 조교수. 관심분야는 네트워킹, 그리드, 모바일 컴퓨팅, 유비쿼트스, 정보보호

추 현 승

1988년 성균관대학교 수학과 졸업(학사) 1990년 University of Texas 컴퓨터공학과 졸업(석사). 1996년 University of Texas 컴퓨터공학과 졸업(박사). 1997년 특허청 심사4국 컴퓨터심사담당관실(사무관). 1998년~현재 성균관대학교 정보통신공학부 부교수. 2001년~현재 한국인터넷정보학회/한국시뮬레이션학회 이사. 2004년 3월~현재 대통령직속 교육혁신위원회 전문위원. 2004년 8월~현재 한국인터넷정보학회 논문지편집위원장. 2005년 1월~현재 건강보험심사평가원 전문위원. 2005년 1월~현재 한국정보과학회 논문지편집위원. 2005년 10월~현재 정보통신부ITRC 지능형HCI융합연구센터장, 정보통신공학부 컨버전스연구소장. 관심분야는 유/무선/광네트워킹, 모바일컴퓨팅, 임베디드SW, 그리드컴퓨팅

조 금 원

1993년 인하대학교 항공우주공학과(학사) 1995년 한국과학기술원 항공우주공학과(석사). 2000년 한국과학기술원 항공우주공학과(박사). 2004년~현재 KISTI e-Science 사업단 응용연구팀장, KISTI 슈퍼컴퓨팅센터 슈퍼컴퓨팅응용지원팀장(겸임). 관심분야는 High Performance Computing, e-Science 및 Grid Computing