초고속 인터넷을 위한 가상 윈도기반의 TCP 성능 개선에 관한 연구

박 형 우† · 정 진 욱††

요 약

최근 인터넷 환경이 반도체, 광통신 그리고 차세대 인터넷 기술의 발달로 고성능화 되어가고 있다. 따라서 고성능 인터넷을 위한 TCP의 성능 향상 연구가 매우 중요해졌다. 그러나 기존 TCP는 수신윈도 버퍼의 물리적 크기에 의하여 최대 전송 성능과 대역폭 탐색 기능이 제한을 받는 구조적인 문제점을 갖고 있다. 본 논문에서는 이를 해결하기 위하여 수신 호스트에 가상 윈도 개념을 도입하였다. 이는 송신 호스트가 RTT 동안 균일하게 세그먼트를 분산시켜서 패킷을 전송할 때 세그먼트 간격 시간 동안 수신 호스트의 처리 능력을 가상윈도로 나타내는 것이다. 따라서 가상 윈도의 크기는 수신 호스트의 성능에 비례하기 때문에 수신 호스트가 고성능일 경우 TCP의 전송 능력 성능이 더 높아질 수 있다. 초고속 인터넷일 경우 제안 알고리즘이 기존 TCP보다 전송능력에 있어 1.5~5배 개선되는 것을 네트워크 시뮬레이터인 NS2를 이용하여 확인하였다.

A Study to Improve TCP Throughput using Virtual Window for Very High Speed Internet

Hyoung-Woo Park† · Jin-Wook Chung††

ABSTRACT

Internet grows rapidly to serve high performance accordingly the technologies of semiconductor, optical communication, and Next Generation Internet evolves. Therefore, the study to enhance TCP efficiencies for high performance becomes very important. The conventional TCP has structural problems of limiting the capability of maximum throughput and the bandwidth probe by their physical buffer size of receive window. This paper introduces "Virtual Window" at receiving hosts to relieve these problems. The Virtual Window represents the throughput of receiving host in processing receiving packets during inter-arrival times of segments when sending host sends TCP segments with regular time intervals within RTT. Consequently, with the proposed algorithm, the better TCP performance for very high speed Internet can be guaranteed if higher performance host is used, with which the bigger size of Virtual Windows is allowed. The proposed algorithm's simulation results using Network Simulator 2 verifies that the throughput of TCP is improved by 1.5~5 times than that of conventional TCP algorithm.

키워드 : 가상윈도(Virtual Window), TCP 혼잡제어(TCP Congestion Control), 고성능 TCP(High Performance TCP), 초고속망(Very High Speed Network), 패킷 스페이싱(Packet Spacing)

1. 서 론

인터넷 환경은 IPOA(IP over ATM), POS(Packet over SONET), WDM (Wavelength Division Multiplexing) 그리고 기가 비트 이더넷 기술 등과 같은 정보통신 기술의 발달과 기가급의 펜티움 칩 개발 등과 같은 반도체 기술의 발달로 빠르게 고성능화 되고 있다. 그리고 인터넷 응용 또한 3차원 가상 현실과 같은 생생한 고품질의 멀티미디어 서비스들이 경쟁적으로 제공되면서 트래픽의 크기가 급속히 증대되고 있다. 이러한 인터넷 응용들은 품질보장을 위하여 TCP에게 고성능으로 전송하여 줄 것을 점점 더 강하게 요구하고 있다.

위와 같이 인터넷 환경이 빠르게 변화되어가고 있지만 현재 대부분의 인터넷 응용이 사용하는 TCP(Transmission Control Protocol)는 1988년에 Jacobson이 혼잡 제어 능력을 보완한 것[3] 외에는 초기 전송 제어 구조[4]를 그대로 유지하고 있다. 현재의 TCP 전송 방식은 AIMD(Additive Increase Multiplicative Decrease) 방식[5]을 기본으로 하고 있다. 즉 송신 윈도를 네트워크 혼잡으로 패킷 손실이 발생할 때까지 선형적으로 늘려가면서 전송하다가, 패킷 손실이 미리 설정된 타이머 등에 의하여 탐지되면 송신 윈도를 반

† 정 회 원 : KISTI 슈퍼컴퓨팅센터 슈퍼컴퓨팅 개발실장/선임연구원
†† 종신회원 : 성균관대학교 전기전자 및 컴퓨터 공학과 교수
논문접수 : 2001년 4월 16일, 심사완료 : 2001년 5월 25일

으로 줄인 후 손실된 패킷을 재전송하고, 다시 선형적으로 송신 윈도를 늘려가면서 전송하는 것을 반복한다. 따라서 TCP의 전송 능력은 전송 속도만으로 결정되는 것이 아니고 수신 윈도의 최대 버퍼 크기, RTT(Round Trip Time) 그리고 네트워크 혼잡 상태 등의 영향을 받는다.

송신윈도는 네트워크 혼잡에 의하여 결정되는 혼잡윈도와 수신 호스트의 수신 가능 윈도 크기중 최소값으로 결정된다. 즉 최대 수신 가능 윈도 크기를 초과할 수는 없다. 따라서 고성능 전송을 위해서 송신 윈도를 증가시키려면 먼저 수신 윈도용 버퍼를 증가시켜야 한다. 즉 수신윈도를 증가시키면 네트워크 혼잡상태가 발생하지 않을 경우 TCP는 송신윈도를 수신가능 윈도의 최대값까지 증가가 가능해지므로 TCP 전송 성능이 높아진다. 그러나 증가 가능한 최대 수신 윈도 크기를 무한정 늘릴 수는 없다. 즉 "전송속도 x RTT"를 초과할 수 없다. 왜냐하면 "전송속도 x RTT"는 수신확인 없이 네트워크 구간을 최대한 채울 수 있는 데이터 전송량을 나타내기 때문이다. 따라서 "전송속도 x RTT"는 TCP를 이용하여 최대한 성능을 얻기 위해서, TCP 송신자와 수신자가 반드시 확보하여야 할 최대 수신 윈도용 버퍼 크기이다.

본 논문에서는 기존의 TCP 수신 윈도의 물리적 크기를 증가하는 방식이 아닌 새로운 가상 윈도 기반 방식으로 수신 윈도를 증가하는 효과를 획득하고자 한다. 즉 수신 윈도의 증대 방식 대신에 만약 수신 호스트의 단위시간당 평균 처리 세그먼트 수를 송신 호스트가 알 수 있게 하여 준다면 송신 호스트는 세그먼트 전송의 간격을 산출하여 해당 시간 동안 수신 호스트가 처리한 세그먼트 크기를 가상의 윈도로 해석하고 가상 윈도 크기만큼 세그먼트를 더 전송할 수가 있다. 이를 위하여 혼잡이 없는 초고속 인터넷 환경일 경우 제안 알고리즘은 TCP 세그먼트를 균분산(even space) 전송하는 기능을 포함하고 있다. 균분산 전송은 TCP 세그먼트가 수신 호스트에 일정 간격으로 도착토록 한다. 그리고 수신 호스트는 헤더 예측[13] 기술을 이용하여 수신된 세그먼트를 도착 순서만 검사한 후 순서가 맞으면 즉시 처리토록 하였다. 따라서 송신 호스트는 "세그먼트 간 휴지 시간 = 도착한 단일 세그먼트 처리 시간"이 될 때까지 송신 윈도를 더 증가 시킬 수 있게 된다. 이때 "세그먼트 간 휴지 시간 = 도착한 단일 세그먼트 처리 시간"일 때의 송신 윈도는 기존 TCP 방식에서 보면 같은 크기의 가상 수신 윈도(V_RWND : Virtual Receive WiNDow)가 있어서 송신윈도가 설정되었다고 해석될 수 있다. 개선 알고리즘은 이러한 가상윈도(V_RWND) 개념을 이용하는데 가상 윈도(V_RWND) 개념을 혼잡제어 알고리즘에 적용하는 것은 세계적으로 처음 시도한 것이다. 기존 알고리즘의 송신 윈도에서는 V_RWND라고 표현하기 어렵다. 왜냐하면 송신윈도 크기는 기존 알고리즘에서 수신윈도 크기 이상으로 설정할 수 없게 정의하고 있기 때문이다. 따라서 송신윈도에 V_RWND를 적용한다는 것은 수신 윈도에 V_RWND를 적용하였을 때만 가능하다. 즉 기존의 송신 윈도는 해당 크기 이상의 물리적 수신 버퍼가 수신 호스트에 존재하는 것을 나타낸다.

그리고 V_RWND의 크기 변화를 송신 호스트에게 알려주는 기능을 또한 제안 알고리즘은 포함하고있다. 4절에 이를 자세히 나타내었다. 제안 알고리즘으로 인하여 개선된 TCP는 혼잡이 발생하지않는 초고속 인터넷 환경에서는 같은 조건의 기존 TCP 보다 1.5~5배 이상의 성능이 향상되었다. 개선된 결과는 NS2(Network Simulator 2)를 이용하여 나타내었다.

2. 초고속 인터넷 환경에서 기존 TCP 윈도의 문제점

초고속 인터넷 환경에서의 기존 TCP 윈도의 문제점을 상세히 분석하기 위하여 먼저 TCP 전송계를 구성하는 요소들이 전송 성능에 미치는 영향을 분석하고자 한다. TCP 전송 요소는 크게 3부분으로 나눌 수 있는데, 송신 호스트, 네트워크 그리고 수신 호스트이다. 이들 3부분은 각각 송신윈도(SWND : Send WiNDow), 혼잡윈도(CWND : Congestion WiNDow), 그리고 수신윈도(RWND : Recivable WiNDow)를 통하여 각각의 성능을 나타낸다. TCP는 송신된 데이터에 대한 수신 확인 신호(acknowledgement)를 수신하면 다시 줄어든 송신윈도를 늘린다. 그리고 분석 편의상 송신 TCP 호스트는 충분히 많은 전송할 데이터를 가지고 있고, 매번 전송할 때마다 1MSS(Maximum Segment Size) 크기의 세그먼트를 SWND 만큼 연속적으로 보낸다고 가정하였다. 식 (1)은 송신 TCP 호스트의 전송 허용된 크기 SWND를 RTT 주기에 대하여 나타낸 것이다.

$$SWND(RTT_n) = \min[\{CWND(RTT_{n-1}) + number\_of\_Ack.s(RTT_{n-1})/K\}, RWND(RTT_{n-1})] \quad (1)$$

$$K = 1, \text{ if } \{SWND(RTT_n) < ssthrsh(RTT_{n-1})\}$$

$$K = SWND(RTT_{n-1}), \text{ if } \{SWND(RTT_n) >= ssthrsh(RTT_{n-1})\}$$

여기서 $number_of_Ack.s(RTT_{n-1})$는 이전 전송 데이터에 대한 수신 확인된 세그먼트 갯수이다. 실제 TCP에서는 세그먼트를 바이트 단위로 표시하지만 여기서는 편의상 RTT 동안 1MSS 크기의 세그먼트를 몇 개 보냈는지를 나타낸다. 패킷의 크기는 1 세그먼트 크기 그대로 전송하는 것으로 가정하였다. 따라서 수신된 Ack. 수는 전송된 세그먼트 수와 같다. 통상 에러가 없었다고 가정하면 $number_of_Ack.s(RTT_{n-1})$은 $SWND(RTT_{n-1})$이다 K = 1은 slow start 모드를 나타내고, $K = SWND(RTT_{n-1})$은 TCP가 혼잡 회피모드를 수행함을 나타낸다. 그리고 ssthresh는 slow start threshold를 나타내며 패킷 손실이 탐지되었을 때의 SWND

값의 1/2이 된다.

식 (1)에서 TCP 송신 시스템의 전송 능력 SWND(RTT_n)는 CWND(RTT_{n-1}), RWND(RTT_{n-1}), 그리고 ssthresh(RTT_{n-1})로 제어되고 있음을 알 수 있다. 이때 RTT_n과 RTT_{n-1}는 n번째와 n-1번째 측정된 RTT(Round Trip Time)을 가리킨다. 따라서 SWND(RTT_n)는 n번째 RTT가 측정될 때의 SWND 값을 가리킨다. 식 (1)을 분석하기 위해서 SWND(RTT_n) = (CWND (RTT_{n-1}) + number_of_Ack.s (RTT_{n-1}) / K)와 SWND (RTT_n) = RWND (RTT_{n-1}) 경우로 나누면 식 (2)와 식 (3)이 된다.

$$SWND(RTT_n) = CWND(RTT_{n-1}) + number\_of\_Ack.s\ (RTT_{n-1}) / K \quad (2)$$

$$K = 1,\ if\ \{SWND(RTT_n) > ssthrsh(RTT_{n-1})\}$$
$$K = SWND(RTT_{n-1}),\ if\ \{SWND(RTT_n) <= ssthrsh(RTT_{n-1})\}$$

$$SWND(RTT_n) = RWND(RTT_{n-1}) \quad (3)$$

식 (2)는 TCP 전송계를 구성하는 송신 호스트, 네트워크 그리고 수신 호스트 중에서 병목현상이 네트워크에서 발생하는 경우이고 식 (3)은 수신 호스트에서 병목현상이 발생하는 경우를 나타내고 있다. 즉 네트워크 대역폭이 충분히 높아서 혼잡이 발생하지 않는 네트워크를 초고속 인터넷이라고 정의하면, 초고속 인터넷에서는 기존의 TCP 수신 호스트의 수신 윈도 크기에 의하여 전송 성능이 결정된다. 현재 일반적으로 TCP 전송시스템에서 기본값(default value)으로 설정하는 수신윈도는 4096 바이트 값을 갖는다. 만약 RTT이 수 밀리초(millisecond)에서 수백 밀리초 범위를 갖는다고 가정할 경우 TCP 전송 성능은 전송망이 충분히 높은 고속의 네트워크일지라도 약 30Mbps(4096 * 8 / 1msec)에서 30Kbps(4096 * 8 / 100msec)의 범위로 제한을 받게 된다. 따라서 수신윈도 값으로 수신 시스템의 성능을 표현하려는 기존 TCP 윈도 방식은 인터넷이 고성능 환경으로 변화될수록 문제점이 커진다. IETF (Internet Engineering Task Force)에서는 관련 문제를 해결하기 위하여 RWND 값을 크게 설정할 수 있는 방법을 RFC 1323[6]에서 권고하고있다. 왜냐하면 TCP 헤더의 16비트보다 더 크게 윈도 크기를 기존 TCP에서 설정할 수 없기 때문이다. 그래서 RFC1323에서는 16비트를 초과하는 수신윈도를 설정할 수 있도록 배수를 TCP 옵션으로 하여 별도로 전송하는 방안을 권고하고 있다.

그러나 윈도(또는 버퍼)의 크기로 호스트의 성능을 정확히 반영할 수 없는 문제점을 여전히 내포하고 있다. 즉 수신 호스트가 300Mhz 급의 PC(Personal Computer)인지 또는 1.5Ghz의 PC 인지에 관계없이 초고속 인터넷 환경에서도 수신 버퍼의 물리적 크기로만 계속적으로 전송 성능을 제어하려 함은 문제가 된다. 왜냐하면 현재 이용하고 있는 경로에 최적화된 수신윈도 크기를 매번 TCP 연결할 때 마다 정확히 알 수가 없고 고속 인터넷 환경에서 사용자들이 임의로 수신윈도 값을 바꾸어 사용할 경우 낮은 수신윈도 사용자는 낮은 성능으로 높은 수신윈도 사용자는 높은 성능으로 초고속 인터넷을 사용할 수밖에 없어서 공정성(fairness) 문제가 더욱 심각하게 발생한다.

초고속 인터넷에서 TCP 윈도 문제점을 해결하기 위한 RFC1323의 권고 방안에는 상기문제 외에도 TCP 헤더의 16비트 윈도 크기를 초과하는 수신윈도를 상대방에게 알리는 방법 또한 문제가 있다. 왜냐하면 RFC1323에서는 시스템 관리자가 적절한 수신윈도 크기를RWND를 판단하여 설정한 후, 이를 TCP 옵션에 실어 보내어 송신 호스트에게 알리도록 규정하고 있다. 이러한 방식은 고성능의 인터넷 환경이 보편화되면 될수록 일반적으로 적용하기 매우 어렵다. 왜냐하면 인터넷 자체가 하나의 블랙박스이고 망 상태가 수시로 변하기 때문에 최적의 통신용 메모리 크기를 실시간으로 판단하고 설정하기가 어렵기 때문이다. 따라서 윈도(또는 버퍼)를 최대한 크게 설정하도록 할 수밖에 없는데, 이는 인터넷이 고성능화 되면 될수록 수신 윈도를 위한 시스템 메모리를 더욱 더 많이 요구하게 되는 결과를 초래한다.

마지막으로 기존 TCP 윈도 방식은 초고속 인터넷일 경우 호스트의 전송 지연 시간이 TCP 전송 성능에 미치는 영향이 무시할 수 없을 정도로 커지는 문제점을 갖고있다. 이로 인하여 기존 TCP의 RTT 시간에 대한 해석의 모호성이 발생한다. TCP 전송계의 전체 지연시간 RTT 중 네트워크에 의한 지연시간을 T_N, 호스트에 의한 지연시간을 T_H 라고 하면 $RTT = T_N + T_H$ 이다. 그리고 네트워크의 속도가 호스트의 물리적 전송 속도 보다 높아서 혼잡(congestion)과 큐잉(queuing)이 발생하지 않을 경우, 호스트에 의한 지연시간을 고성능의 호스트에 의해서 발생된 지연 시간 $T_{H,H}$와 낮은 성능의 호스트에 의하여 발생된 지연시간을 $T_{H,L}$로 나누어 표기하자. 그러면 $T_{H,H} <= T_{H,L}$ 이 된다 두 호스트가 같은 조건의 초고속 네트워크이라면 $T_N + T_{H,H} <= T_N + T_{H,L}$ 이다. 따라서 "대역폭 x RTT"는 식 (4)와 같이 표현될 수 있다.

$$B * (T_N + T_{H,H}) <= B * (T_N + T_{H,L}) \quad (4)$$

식 (4)을 들여다보면 고성능 인터넷 환경에서는 TCP 성능을 최대화 시키기 위해서 "대역폭 x 전송지연시간(RTT)" 만큼 TCP 호스트의 송수신 버퍼를 증대시키는 방안이 효율적이지 못함을 알 수 있다. 상대적으로 낮은 속도의 네트워크가 보편적이던 초기 인터넷 시대에는 RTT 시간에 포함되어있는 수신 호스트의 지연 시간은 네트워크에 의한 전송 지연 시간에 비하여 상대적으로 작아서 무시될 수 있

지만, 최근 초고속망의 보급으로 네트워크에 의한 전송 지연시간이 상대적으로 크게 줄어들었고 호스트에 의한 전송 지연과 초고속 대역폭과의 곱한 값이 상대적으로 무시할 수 없을 정도로 커졌다. 따라서 고성능을 위한 TCP 전송 알고리즘은 수신호스트의 성능 변화도 효율적으로 포함할 수 있도록 TCP 윈도의 개념도 개선되어야 한다.

3. 관련 연구

TCP 성능 향상을 위한 연구는 1990년대 초부터 슈퍼컴퓨터 네트워크와 같은 고성능의 전산 자원간의 통신이 필요한 특수 R&D 네트워크 커뮤니티를 중심으로 연구되어왔다. 따라서 이들 연구는 오늘날 초고속망이 널리 보급되면서 TCP의 고성능화 연구의 길잡이가 되고있다. 이들의 연구 방향은 크게 4가지로 나눌 수 있다. 첫 번째는 송신 호스트가 path MTU(Maximum Transmission Unit) Discovery 기능을 활용하여 최대 path MTU의 크기를 인지토록 한 후 IP가 최대 path MTU 패킷을 전송할 수 있도록 TCP 세그먼트 크기를 재설정 하는 것이다[7]. 두 번째는 TCP의 전송 윈도의 크기를 증가시키는 것이다. 이는 초고속 인터넷에서 고성능 TCP 전송을 위하여 호스트가 수신확인 없이 보낼 수 있는 세그먼트 전송량을 늘리도록 한 가장 많이 쓰이는 방안이다[6, 8]. 세 번째로는 호스트 시스템이 충분히 큰 소켓 버퍼를 갖도록 하거나 또는 응용 프로그램 자신이 송수신용 소켓 버퍼를 전송 링크의 bandwidth * delay 크기를 갖도록 하는 것이다[6, 8]. 이는 두 번째 방법과도 관련이 있다. 마지막 네 번째로는 TCP Selective Acknowledgements(SACK)을 이용하여 재전송 횟수를 줄이는 것이다[9]. 이는 초고속망의 경우 기존의 축적형 수신확인(cumulative acknowledgements)을 사용할 경우 다량의 데이터가 재전송하게 되면 전송 효율이 저속의 네트워크에 비하여 급격히 떨어질 수 있기 때문이다. 정리하면 초고속 망을 위한 TCP 고성능화 연구는 주로 수신호스트의 통신용 송수신 버퍼를 크게 한 후, 송수신 윈도가 큰 값을 갖도록 TCP 호스트의 관련 시스템 파라메터를 수정함으로써 고성능의 전송을 구현하는 방법들을 제시하고 있다.

상기와 같이 여러 가지 방법들이 연구되고 있지만, 기존 TCP가 사용하는 고정된 윈도 (특히 수신 버퍼 크기) 기반 전송시스템이 갖는 문제점에서 근본적으로 벗어나지 못하고있다. 즉 호스트의 성능을 충분히 반영치 못하는 문제, 메모리 과다 설정, 그리고 시스템 관리자에 의한 수신윈도 값의 수동 설정에 의한 비효율성 문제는 여전히 남아있다.

따라서 본 연구에서는 기존 TCP 전송 시스템의 기본 골격인 윈도 기반 전송 시스템을 그대로 유지하면서도 윈도 크기의 고정성을 극복하고 수신 호스트의 성능을 충분히 반영할 수 있는 가상 윈도를 새롭게 도입하였다. 가상윈도 도입으로 제안 알고리즘은 수신윈도 한계성을 극복할 수 있어서 수신윈도 이상으로 송신 윈도의 증가를 허용한다. 수신윈도 값을 넘어서 전송할 경우 제안 알고리즘은 TCP 세그먼트를 균분산(even space) 전송하도록 한다. 균분산은 TCP 세그먼트가 수신 호스트에 일정 간격으로 도착토록 한다.

제안 앨고리즘에서 사용하는 세그먼트 균분산 전송 기법은 쉘 스페이싱(Cell Spacing)[10]과 Leaky Bucket[16] 알고리즘과 유사한 것 같지만 전혀 다르다. 이들 기법들은 트래픽 제어기술로서 평균 속도와 최고속도가 사용자와 서비스 제공자에 의하여 사전에 계약을 통하여 정해지면, 네트워크 서비스 동안 평균 속도와 최고속도가 그대로 고정된 상태로 적용되어 사용자가 서비스 보장을 받을 수 있게 한다. 본 제안 알고리즘은 블랙 박스와 같은 인터넷의 상황에 동적으로 최적 적응할 수 있도록 최대치와 평균치가 변화하도록 하였다. 기존 TCP의 AIMD 방식에 의하여 네트워크의 혼잡 상태나 수신호스트의 성능에 따라 윈도의 최대치가 탐색되면, 균분산 전송을 통하여 RTT 동안 TCP가 탐색된 최대치에 기반하여 평균치 전송을 하도록 하는 것이 다르다. 그리고 이러한 윈도 값 범위 내에서 속도 개념을 도입하는 것은 기존에 연구되고 있는 속도기반 혼잡제어 알고리즘 연구와 다르다[11, 12]. 왜냐하면 속도기반 혼잡 제어 알고리즘은 TCP 전송 속도를 혼잡상태에 따라 제어하여 패킷 손실이 발생하지 않도록 한다. 따라서 속도기반 혼잡제어 알고리즘들은 공통적으로 인터넷의 혼잡 상태를 현실적으로 정확하게 측정하는 것이 불가능하기 때문에 TCP 전송 속도가 네트워크에서 전송 가능한 실제 속도보다 낮게 결정되는 경우가 많은 문제점을 갖는다. 본 논문에서 제안하는 알고리즘은 기존의 TCP의 윈도기반 제어 알고리즘을 유지하기 때문에 전송 가능한 대역폭 탐색 기능이 계속적으로 유효하게 수행된다.

4. 제안 알고리즘

앞에서 지적했듯이 기존 TCP는 혼잡이 발생하지 않는 네트워크 즉 초고속 망일 경우에는 수신 윈도 크기 이상으로 전송을 할 수 없는 근본적인 문제점을 갖고있다. 따라서 해결 방안으로 수신윈도 크기를 확대하는 방향으로 연구들이 진행되어 왔는데 이는 기존 TCP 윈도를 매우 단순하게 확대하는 방식이기 때문에 기존의 문제점을 여전히 내포하고있다. 더욱이 윈도 확장에 기반한 해결책은 윈도 크기 차이에 의한 공정성 문제를 새롭게 초래한다. 왜냐하면 혼잡이 없는 네트워크에서는 전송 성능이 윈도의 크기에 비례하기 때문이다. 즉 글로벌 인터넷에서 통일된 윈도크기를 네트워크 상황에 따라 일일이 권고하기는 매우 어렵다. 본 논문에서는 이러한 문제점들을 극복하면서 동시에 초고속

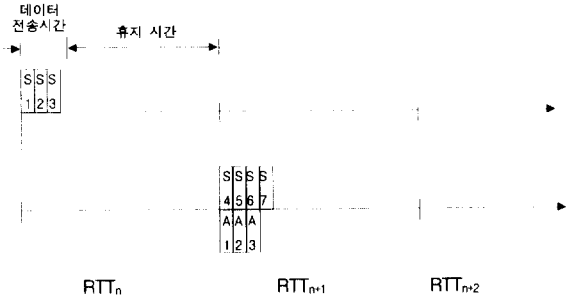
망에서도 공정성을 지원할 수 있는 가상윈도 기반 TCP 성능 향상 알고리즘을 제시하고자 한다. 즉 새로 제안하는 알고리즘은 고속 모드를 위한 것이기 때문에 SWND >= RWND를 고속모드로 새롭게 정의하고 이때에만 동작되도록 설계되었다. SWND < RWND에서는 기존 TCP의 기능만 수행한다. 따라서 기존 TCP와 100% 호환성을 유지한다. 정리하면 제안 알고리즘은 slow start 모드와 혼잡 회피 모드로만 구성되어 있는 기존의 TCP 전송제어 모드에 고속 모드를 위한 알고리즘을 추가하는 형태로 기존 TCP에 접목된다.

제안 알고리즘의 구성은 3부분으로 되어있다. 첫 번째는 전송하고자 하는 TCP 세그먼트를 RTT 시간 동안에 균일하게 분산하여 전송하는 부분이다. 이는 단일 RTT내에서 최대치 기반 윈도 제어 방식을 사용하는 기존 TCP 전송 방식에 가상 윈도 개념을 도입하기 위하여 평균치 기반 속도 제어 방식으로 TCP 전송 방식을 변환한다. 두 번째는 수신 호스트의 성능 정보까지 수신윈도 값에 포함시킬 수 있는 가상 수신 윈도(V_RWND)의 환경에서 단대단 윈도 시그날링에 대한 부분이다. 마지막 세 번째는 가상 윈도 환경에서 slow start 모드, 혼잡회피 모드 그리고 고속모드로 식별되어지는 환경에서 가상 윈도의 슬라이딩 처리에 대한 부분이다. 즉 네트워크 전송속도, 송수신 호스트의 성능에 따라서 slow start 모드, 혼잡회피 모드 그리고 고속모드의 크기가 동적으로 변화하도록 하였다.

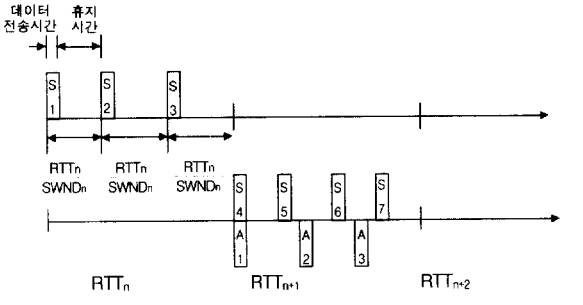
4.1 세그먼트 균분산 전송하기

기존 TCP에서 트래픽 발생 모습은 (그림 1)과 같이 나타낼 수 있다. 이때 TCP는 전송할 데이터를 충분히 많이 갖고 있다고 가정하였다. 따라서 네트워크가 허용한 윈도 크기만큼은 언제든지 보낼 수 있다. 그리고 패킷의 크기는 모두 1 MSS(Maximum Segment Size)의 패킷 크기로 네트워크 상에서 전송되는 것으로 가정하였다. (그림 1)은 송신윈도(SWND)가 3인 때의 TCP 전송 상태를 나타내고, RTTn은 n번째 RTT 값을, 약어 S는 세그먼트, 그리고 약어 A는 수신확인(acknowledgment)을 나타낸다. 혼잡이 없는 초고속 인터넷에서, TCP는 (그림 1)과 같이 최대 허용 윈도 크기의 데이터를 일시에 보내고, 대부분의 RTT 시간을 기다린다. 실제 인터넷에서 TCP 전송은 Ack. 신호를 수신할 때마다 윈도 값이 증가되고, 전송하고자 하는 데이터 또한 임의적으로 응용에서 발생하므로, RTT구간 동안 데이터 전송시간 T_d와 휴지시간 T_i는 임의적으로 분산되어 나타날 수 있다. (그림 1)은 최고로 집중된 경우를 나타낸 것이다.

제안 알고리즘에서 세그먼트 균분산 전송하기는 임의의 RTTn 동안에 데이터를 균일하게 분산시켜서 전송하는 것을 보장하는 기법으로 일종의 동적 적응형 트래픽 스페이싱 기법이다[1, 2]. (그림 2)에 세그먼트 균분산 전송하기를 개념적으로 나타내었다. (그림 2)와 같이 데이터 전송시간과 휴지시간을 균일하게 분포 시킴으로써 수신호스트의 세그먼트 처리성능에 비례하는 TCP 전송이 가능해진다. RTT 동안의 데이터 송신 호스트의 전송 속도는 SWND/RTT를 이용하여 구할 수 있다. 즉 TCP는 SWND/RTT 이상의 속도로 전송을 하지않는다. 해당 TCP 세션에서 최대 폭주 속도는 SWND/RTT이다 라는 것과 같은 표현이다.



(그림 1) 기존 TCP의 트래픽 생성



(그림 2) 균분산 트래픽 전송

이러한 균분산 전송 모드에서는 기존 윈도 값들을 <표 1>과 같이 새롭게 해석할 수 있다. <표 1>에서 파라메터 단위 중 sps는 segments per second를, n은 변화분이 0이 아닌 임의의 수를 나타낸다.

〈표 1〉 윈도의 새로운 의미

기존값[단위]	균분산 후	새로운 의미
SWND[byte]	SWND/ RTT[sps]*	송신속도
CWND[byte]	CWND/ RTT[sps]	네트워크 허용 속도
RWND[byte]	직전 RWND 값의 차이[n]	수신호스트의 패킷 처리 능력 비교

4.2 가상 수신윈도(V_RWND) 기반 단대단 윈도 시그날링

기존 TCP에서는 수신된 패킷들에 대해서 수신 호스트가 송신 호스트에게 수신 확인(Ack.) 신호를 보낼 때, TCP 헤더의 수신 윈도 필드에 수신 호스트가 수신 가능한 윈도(RWND) 크기 정보를 삽입하여 보낸다. 이때 균분산 전송 환경에서는 수신 호스트가 수신 버퍼에서 패킷 처리하는 속도보다 빠르게 패킷이 도달할 경우 수신 가능한 윈도의 크기는 작아지게 된다. 즉 $RWND_{n+1} < RWND_n$ 으로 표현된다. 만약 수신 호스트가 패킷의 수신되는 속도보다 빠르게 처리

한다면 수신 가능한 윈도의 크기는 항상 같을 것이다. 그 크기는 시스템 관리자가 초기 설치한 크기 그대로가 된다. 즉 $RWND_{n+1} = RWND_n$ 으로 표현되고 그 값은 $RWND_{max}$ 가 된다.

따라서 균분산 전송 환경에서는 송신 호스트는 이전에 Ack. 신호와 함께 수신된 RWNDn 값을 보관하고 나서, 바로 다음 Ack. 신호와 함께 수신된 $RWND_{n+1}$를 비교하면 수신 호스트의 처리 성능을 비교할 수 있다. 즉 송신 호스트는 비교값을 기준으로 송신 윈도를 증가시키거나 감소시킬 수 있다. 예를 들면, RWND 값이 직전 수신된 RWND 보다 값이 같으면(또는 초기값을 나타내면) 송신 호스트는 진행중인 전송모드(slow start 모드, 혼잡 회피모드 또는 고속모드)에 따라서 송신 윈도를 증가시킨다. 만약 RWND 값이 직전에 수신된 RWND 값보다 작으면 송신 호스트가 수신 호스트의 처리속도보다 점점 더 빠르게 세그먼트를 전송하고 있음을 판단할 수 있으므로 송신 호스트는 전송모드를 혼잡회피 모드로 전환하게 한다.

이때 RWND의 변화를 송신 호스트가 알기 위해서는 RTT/2 정도가 소요된다. 초고속망일 경우에는 이 시간 동안 많은 데이터가 전송되어져 패킷 손실이 발생될 수 있는데, 제안 알고리즘에서는 이럴 경우 기존 TCP와 똑같이 재전송 타이머를 이용하여 패킷 손실을 탐지한다. 정리하면 전술한 바와 같이 V_RWND는 "세그먼트 간 휴지 시간 = 도착한 단일 세그먼트 처리 시간" 일 때의 송신 윈도로 설정되며 식 (5)와 같이 나타낼 수 있다.

$$\mathbf{V\_RWND = RWND + (RTT/SWND) * \Delta T_p} \qquad (5)$$

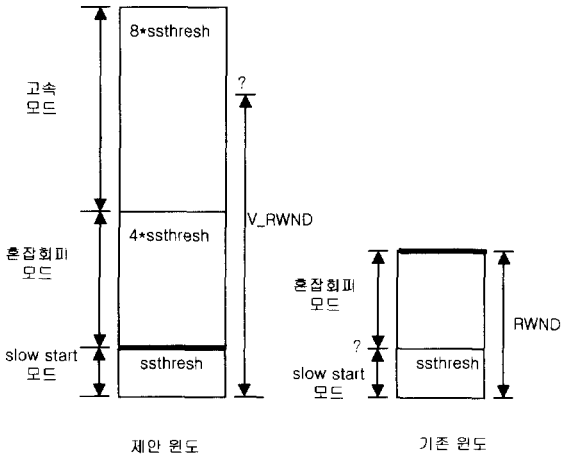
여기서 ΔT_p는 수신 호스트의 바이트 단위의 초당 패킷 처리 능력을 가리킨다.

4.3 가상 윈도 슬라이딩 범위 계산

위에서 정의된 V_RWND는 호스트의 전송 속도, 네트워크 대역폭 그리고 수신 호스트의 성능 변화에 따라서 바뀐다. 그러므로 기존 TCP처럼 가상 수신 윈도의 크기를 최대 송신 가능한 윈도로 설정되기 때문에 기존 TCP와 같이 slow start 모드, 혼잡회피 모드 그리고 고속 모드의 범위를 가상 수신 윈도 환경에 맞게 새롭게 정의하여야 한다. 기존 TCP는 네트워크 혼잡에 의한 설정된 ssthresh를 기준으로 각 모드를 식별 하였다. 따라서 제안 알고리즘도 기존 TCP와 같이 패킷 손실에 의하여 설정된 ssthresh를 기준으로 slow start 모드 구간, 혼잡 회피 모드 구간 그리고 고속 모드 구간을 정의하였다.

인터넷 트래픽 모델에 대한 최근의 연구는 인터넷 트래픽이 self-similar 모델의 특징을 갖는 것으로 연구되고 있다.

self-similar 모델의 가장 큰 특징은 트래픽의 분산 값이 샘플링 시간의 크고 작음에 관계없이 3~4의 값을 갖는다



(그림 3) 가상 윈도 슬라이딩 범위 계산

[15]. 이는 8 * ssthresh까지 송신 윈도를 증가시키면 네트워크 혼잡 또는 송신 호스트와 수신 호스트의 성능 차이에 의하여 반드시 패킷 손실이 1회이상 발생한다는 것을 나타낸다. 즉 송신 윈도 값을 1에서 8 * ssthresh까지 증가시키면 패킷 손실이 적어도 1회이상 발생하기 때문에 제안 알고리즘은 임의의 초고속 속도를 갖는 초고속 인터넷 환경에서도 동작될 수 있다. 따라서 고속모드의 한계를 (그림 3)과 같이 8 * ssthresh로 크기로 제한할 수 있다. 그리고 제안 알고리즘에서는 기존 알고리즘의 slow start 모드와 혼잡회피 모드에 대한 기존의 정의를 인용하였다. 그러나 혼잡회피 모드의 한계값(또는 고속 모드의 시작값)은 새롭게 정의하였다. 왜냐하면 기존 TCP에서는 혼잡회피 모드의 한계값을 단순히 RWND 크기로 정의하였기 때문이다. 제안 알고리즘에서는 송신 윈도의 한계 범위를 고속모드의 크기로 정의하였는데, 고속모드에서의 TCP 역할은 동적으로 변하는 고속 인터넷의 상태를 새롭게 탐색하는 부분이다. 탐색시간을 짧게하기 위하여 1 RTT 동안 탐색행위가 이루어 질 수 있도록 slow start 모드처럼 윈도 증가를 지수 함수적으로 증가시키도록 설계하였다. 따라서 1 RTT 동안 8* ssthresh에 도달할 수 있는 4 * ssthresh 까지를 혼잡회피 모드의 한계로 제안 알고리즘에서 정의하였다.

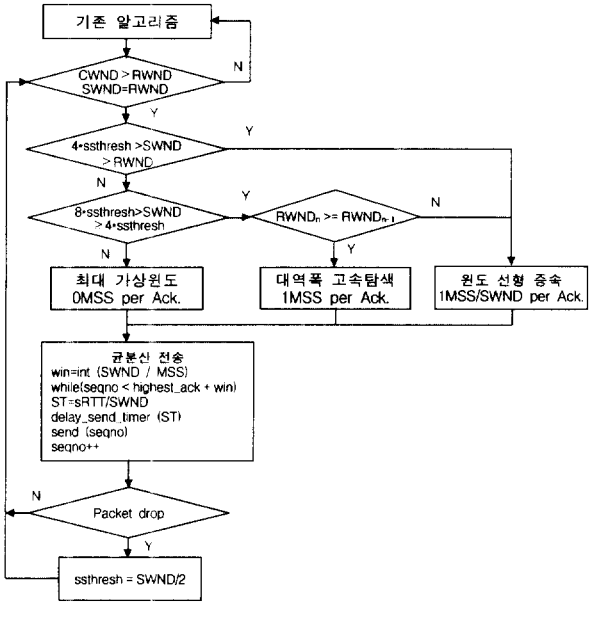
그리고 시뮬레이션을 통하여 확인하였지만 새롭게 정의된 혼잡회피 모드 범위 내에서(1 * ssthresh ~ 4 * ssthresh) 송신 호스트는 패킷 손실을 경험하였다. 즉 고속모드에서 패킷 손실을 탐지하는 경우는 거의 작았다. 제안 알고리즘에서도 혼잡 회피 모드에서는 기존 알고리즘과 같이 송신 윈도 증가 속도를 1 RTT 마다 1/CWND 만큼 증가한다. 정리하면 제안 알고리즘은 RWND 이상 송신 윈도를 증가시킬 수는 있지만 8 * ssthresh 를 초과할 수는 없다. 이는 균분산 전송으로 인하여 도입된 전송 지연 시간으로 인하여 공정성이 악화되는 것을 방지하기 위함이다. 왜냐하면 TCP에서 혼잡이 발생할 때까지의 윈도 증가는 모든 TCP 세션이 같은 수준의 대역폭을 점유할 수 있도록 같은 정도의 윈도 증가

노력을 할수있도록 하여야 하기 때문이다. 따라서 패킷 손실이 발생이 보장되는 범위에서 최소값을 고속모드의 한계치로 설정하였다.

4.4 제안 알고리즘 구조

제안 알고리즘은 (그림 4)와같다. NS2 시뮬레이터의 TCP와 TCP Reno 소스를 수정하여 구현하였다. 수정된 부분은 고속 모드일 경우 트래픽을 균분산 전송 하고, 가상 윈도를 정의하고 가상윈도 기반 전송 제어, 그리고 모드별 가상 윈도의 슬라이딩 크기 조정이다. 시뮬레이션에서 비교할 TCP로 TCPreno를 선택한 이유는 TCPreno가 매우 널리 사용되는 TCP 형태이기 때문이다. TCP reno는 fast recovery 와 fast retransmit 기능을 제공한다.

(그림 4)에서 seqno는 TCP 세그먼트의 시퀀스 번호를 나타내며 delay_send_timer는 세그먼트간 스페이싱 시간을 제어하기 의하여 추가되는 TCP 타이머이다. RTT는 기존 알고리즘에서 추출된 값을 활용한다.

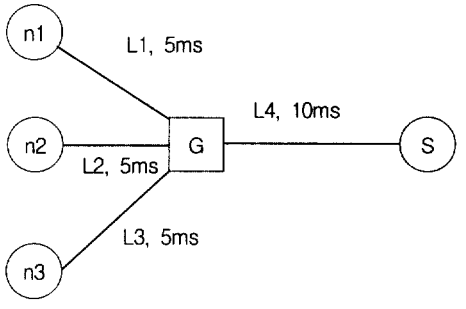


(그림 4) 제안 알고리즘

5. 시뮬레이션 및 결과

고성능 전송을 위한 가상윈도 기반 TCP 성능 개선 알고리즘을 기존 TCP 알고리즘과 비교하기위하여 시뮬레이션 토폴로지를 NS2[14] 시뮬레이터를 이용하여 다음과 같이 구성하였다. 노드 n1과 n2는 TCP 노드로서 제안 알고리즘과 기존 알고리즘과의 성능 비교와 페아니스를 비교하기 위하여 2개 노드를 설정했다. n1과 n2 모두 응용으로는 FTP를 선정하였으며 전송할 데이터 무한대로 있는 것으로 설정하였다. n3는 UDP(User Data Gram) 노드로서 링크 L4의 백그라운드 트래픽을 위하여 설정하였으며 응용은 NS2에서 정의한 기본 값으로 정의한 CBR(Constant Bit Rate) 응용으로 설정하였다. 노드 n1, n2 그리고 n3의 응용을 CBR 형태로 설정한 것은 제안 알고리즘으로 인한 성능 개선 효과를 최대한 나타내기 위함이다. VBR을 사용하였을 경우에도 개선 효과가 나타나지만 CBR만큼 크게 나타나지 않는다. 왜냐하면 VBR로서 수신윈도의 크기 제한에 의한 성능 저하를 표현하기가 상대적으로 어렵기 때문이다.

NS2의 소스는 참고문헌 14번에서 획득하였으며 TCP Reno 소스는 NS2의 소스에 포함되어있는데 TCP.cc 와 TCPReno.cc로 구성된다. 왜냐하면 NS2은 객체지향 프로그램 기법으로 작성되어있기 때문입니다. 그리고 TCP Reno를 비교대상으로 선정한 이유는 현재의 인터넷 상에서 가장 많이 사용하고 있는 TCP 버전으로서 fast retransmission 과 fast recovery 기능을 지원합니다.



(그림 5) 시뮬레이션 토폴로지

시뮬레이션에서 노드 n1과 n2의 속도를 똑같이 설정하였다. 이는 개선된 알고리즘과 기존 알고리즘의 성능과 공정성을 비교하기 위함이다. 그리고 노드 n1과 n2의 속도가 10Mbps일 때와 45Mbps일 때 두 가지 경우에 대하여 L4의 속도를 2Mbps~622Mbps로 변화시켜가면서 노드 n1과 n2의 전송 성능을 측정하였다. 이때 기존 TCP의 윈도크기가 TCP 성능에 미치는 영향을 분석하기 위하여 n2의 수신윈도는 8MSS로 고정시켜놓고, n1의 수신윈도를 MSS의 8배, 16배, 32배, 그리고 64배로 변화시켜가면서 TCP노드 n1과 n2의 성능을 측정하였다. 고속모드가 주 실험 대상임에도 불구하고 저속의 링크에서도 실험을 한 것은 제안 알고리즘이 기존 알고리즘과 호환성을 측정하기 위함이다. 시뮬레이션에서 MSS는 1000바이트로 설정하였다. 그리고 관련 파라미터를 <표 2>와 <표 3>에 정리하였다.

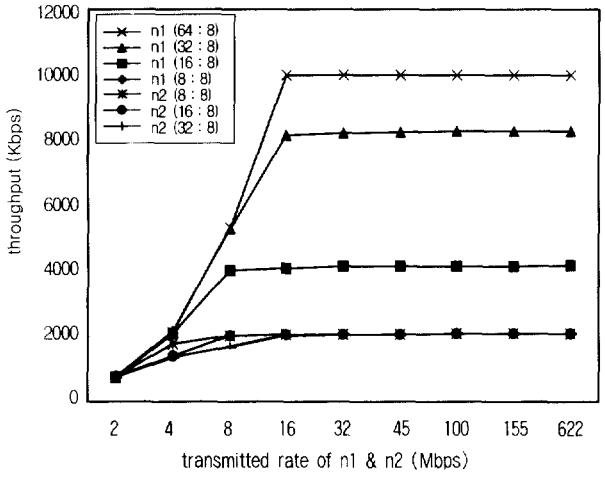
〈표 2〉 노드 파라미터

파라미터	노드 n1	노드 n2	노드 n3	조건
응 용	FTP	FTP	FTP	무한 데이터
프로토콜	TCP	TCP	UDP	
수신 윈도	8,16,32,64 MSS로 가변	8MSS로 고정	3.75ms 주기로 210 바이트 패킷 전송	

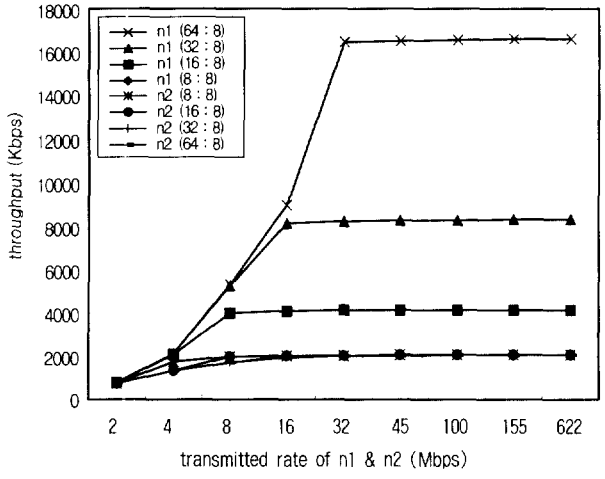
〈표 3〉 링크 파라미터

파라미터	링크 L1	링크 L2	링크 L3	링크 L4	비고
전송 지연	5 ms	5 ms	5 ms	10 ms	
링크 형태	drop tail	drop tail	drop tail	RED	
링크 속도	10 또는 45Mbps	10 또는 45Mbps	10Mbps	2~622Mbps 로 가변	
버퍼 크기	None	none	none	20	단위 : MSS

(그림 6)은 n1과 n2의 속도가 10Mbps일 때의 시뮬레이션 결과이고, (그림 7)은 n1과 n2의 속도가 45Mbps일 때의 시뮬레이션 결과이다. (그림 6)과 (그림 7)에서 혼잡이 발생하지 않는 초고속 인터넷에서는 TCP 성능은 식 (3)과 수신윈도의 크기에 비례하고 있음을 알 수 있다. (그림 6)과 (그림 7)에서 표기 n1(64 : 8)은 TCP 노드 n1과 n2의 수신윈도가 64MSS와 8MSS 크기일 때의 시뮬레이션 결과인 노드 n1의 성능이며, n2(64 : 8)는 같은 시뮬레이션으로서 TCP 노드 n1과 n2의 수신윈도가 64MSS와 8MSS 크기일 때의 시뮬레이션 결과인 노드 n2의 성능을 나타낸다. (그림 6)에서 n1(64 : 8)의 성능이 10Mbps을 넘을 수 없음은 (그림 6)



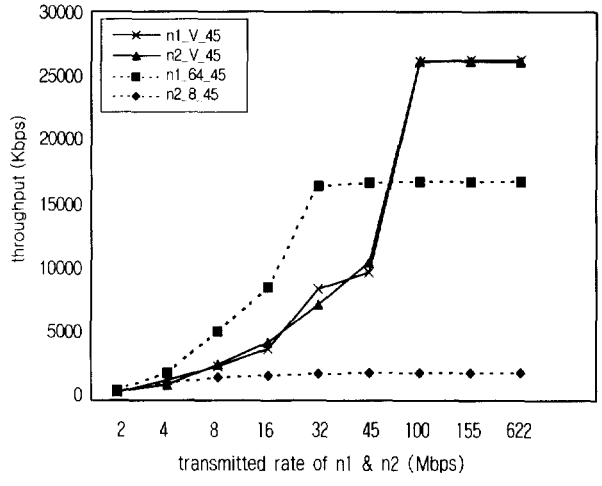
(그림 6) 수신윈도 크기 대비 성능비교 1



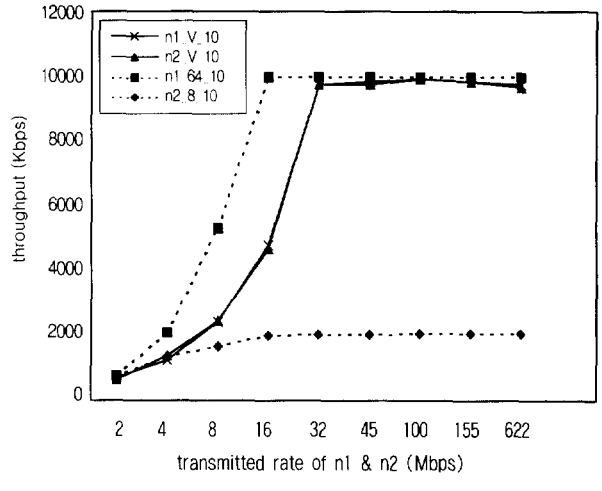
(그림 7) 수신윈도 크기 대비 성능비교 2

이 노드 n1과 n2의 전송 인터페이스 속도가 10Mbps이기 때문에 수신 윈도를 최대로 증가시킬 경우 최고 10Mbps 전송 성능에 도달할 수 있다. 본 논문에서는 수신호스트의 세그먼트 처리 속도를 송신 호스트의 물리적 전송속도와 같게 설정하였다.

이는 가상윈도로 인한 성능 개선 효과를 최대로 보여주기 위함이다. (그림 6)과 (그림 7)에서 n1과 n2의 수신윈도 크기가 8MSS인 경우가 다섯번 있는데 이들 성능 곡선이 서로 겹쳐서 그래프상에서 하나하나 구별되어 보이지 않고 있다.



(그림 8) 가상윈도에 의한 성능 개선 효과 1

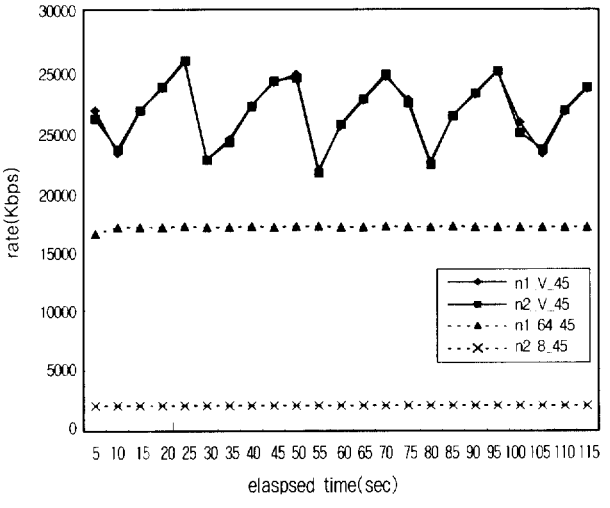


(그림 9) 가상윈도에 의한 개선 효과 2

(그림 8)과 (그림 9)는 노드 n1과 n2가 초고속 인터넷일 경우 가상윈도가 동작하도록 설정한 시뮬레이션의 결과를 보여준다. 노드 n1과 n2 모두 물리적 수신윈도 크기를 모두 8MSS로 설정하였다. 이때 (그림 8)은 n1과 n2의 전송속도가 45Mbps 인 경우이고 (그림 9)는 n1과 n2의 전송속도가 10Mbps 일 때의 시뮬레이션 결과를 나타낸다. (그림 8)과 (그림 9)에서 알 수 있듯이 가상윈도에 의하여 성능 개선이

같은 수신윈도 크기와 비교할 경우 1.5~5배 이상 개선될 수 있음을 나타내고 있다. (그림 8)과 (그림 9)에서 표기 n1_V_45는 노드 n1이 가상윈도를 사용하고 링크속도가 45 Mbps일 때를 가리킨다. 그리고 n1_64_45는 같은 방식으로 노드 n1이 수신윈도가 64 MSS이고 전송속도가 45Mbps인 것을 나타낸다. 그리고 n1_V_45와 n2_V_45일 때의 시뮬레이션 결과와 n1_64_45와 n2_64_45일 때의 시뮬레이션 결과를 서로 비교하기 위하여 한 그래프에 나타내었다.

(그림 8)과 (그림 9)에서 알 수 있듯이 초고속 인터넷 환경에서 가상윈도를 도입하였을 경우 공정성도 크게 개선됨을 알 수 있다. 시뮬레이션에서 수신 호스트들의 성능을 같게 하여서 개선된 것으로 해석될 수도 있지만 수신 호스트의 성능에 비례하는 가상윈도 개념 도입으로 패킷 손실을 유발시킬 수 있도록 전송능력의 확대가 가능하기 때문에 초고속 인터넷에서도 공정성 개선 노력이 지속적으로 유효하게 동작할 수 있다. 왜냐하면 기존 TCP의 공정성 유지 수단이 패킷 손실을 유발시키는 윈도 증가에 의한 대역폭 탐색 방법이기 때문이다.



(그림 10) 5초간의 평균속도 변화

(그림 10)은 초고속 인터넷에서도 가상윈도를 도입하면 송신윈도의 증가 모습이 마치 저속모드와 같이 톱니형 모습을 보여주고 있음을 알 수 있다. 이는 가상윈도의 최대값이 8 * ssthresh로서 송신윈도가 8 * ssthresh에 도달할 때까지 증가시키면 반드시 패킷 손실이 발생하는 최소값이기 때문이다.

6. 결 론

TCP 윈도에 수신 호스트의 성능에 비례하는 크기를 갖는 가상윈도를 도입하여, 초고속 인터넷에서도 수신 윈도의 물리적 크기의 한계 없이 기존 TCP가 동적으로 대역폭을 지속적으로 탐색할 수 있게 하였다. 따라서 TCP성능을 초고속 인터넷 환경일 경우 크게 향상시킬 수 있게 하였다. 가상윈도는 기존 TCP가 수신윈도의 물리적 버퍼 크기에 의존하였을 경우 나타나는 구조적인 문제 - 윈도기반 불공정성(unfairness), 고성능 전송을 위한 메모리 과다 확보, 그리고 수동식 수신윈도 설정 등 - 를 근본적으로 해결할 수 있는 방법을 제시하였다. 따라서 향후 초고속 인터넷 시대에서도 베스트 이포트 서비스가 지속적으로 유효하게 서비스될 수 있는 길을 열었다고 판단한다. 가상윈도 개념을 위하여 도입된 TCP 트래픽을 균분산 전송은 패킷들이 전송도중 도착순서가 바뀌는 확률을 낮출 수 있다. 즉 RTT/(윈도 + 1) 만큼의 범위 내에서 앞선 패킷이 지연되어도 패킷의 도착순서가 바뀌지 않는다. 즉 header prediction 효율을 높일 수 있어서 고성능 인터넷 전송 환경에서의 TCP전송 성능을 또한 높일 수 있다.

가상 윈도를 설정함으로써 송신 호스트가 물리적 수신 윈도 크기 이상으로 데이터 전송이 가능하기 때문에 성능이 개선되지만 이러한 성능 개선은 항상 이루어지는 것이 아니고 다음 두 가지 조건이 만족할 때 성능 향상이 이루어 진다. 첫째 전송 대역폭이 고속이어서 혼잡이 발생하지 않을 때, 둘째 전송 대역폭이 수신 호스트의 수신 패킷 처리 속도보다 더 빠를 때이다. 따라서 상기 조건이 일반 인터넷에선 발생하기 어렵지만 패스 이더넷(fast ethernet) 또는 기가 이더넷(giga ethernet)을 사용하는 고속 캠퍼스 네트워크나 슈퍼컴퓨터 네트워크와 같은 특수 연구 목적용 초고속 네트워크에선 적용이 가능하다. 본 논문에서는 가상윈도의 실험을 시뮬레이션 환경에서 수행하였지만 향후 리눅스의 관련 TCP 커널을 수정하여 실제 인터넷에서 시험하고자 한다.

참 고 문 헌

[1] Hyoung Woo Park, Jin Wook Chung, "A Study on Reduction of Traffic Burstness using Window Based Segment Spacing," Proc. IEEE ICOIN-15 Japan Beppu, pp.41-45, Jan. 2001.

[2] Hyoung Woo Park, Jin Wook Chung, "Improvement of Congestion Control for Burst Traffic," Proc. PDPTA'2000 U.S.A., Las Vegas, pp.1029-1033, Jun. 2000.

[3] V. Jacobson, "Congestion Avoidance and Control," Proc. ACM SIGGOM '88, pp.314-329.

[4] Postel, J., Ed. "Transmission Control Protocol Specification," SRI International, Menlo Park, CA, RFC-793, Sep, 1981.

[5] D. Chiu and R. Jain, "Analysis of the increase and decreas algorithm for congestion avoidance in computer networks," Journal of Computer Networks and ISDN, 17(1) : 1-14, Jun, 1989.

[6] V. Jabcobson, R. Braden, D. Borman, "TCP Extensions for High Performance," RFC-1323, May, 1992.

[7] J. Mogul, S. Deering, "Path MTU Discovery," RFC-1191,

Nov, 1990.

[8] J. Mahdavi, "Enabling High Performance Data Transfers on Hosts," PSC, http : //www.psc.edu/networking/perf_tune.html. Sept. 1999.

[9] M. Mathis, J. Mahdavi, S. Floyd, A. Romanow, "TCP Selective Acknowledgment Options," RFC-2018, Oct. 1996.

[10] P. Boyer, F. Guilleman, M.Servel, J. Coudreuse, "Spacing Cells Protects and Enhances Utilization of ATM Network Links," IEEE Network, Sep, 1992.

[11] R. Rejae, M. Handley, D.Estrin, "RAP : An End-to-end Rate-based Congestion Control Mechanism for Realtime Streams in the Internet," Proceedings of IEEE INFOCOM '99, New York, Mar, 1999.

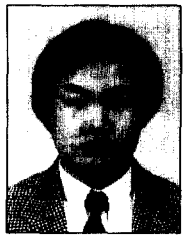
[12] L. S. Brakmo, S. W. O'Malley, "TCP Vegas : New Techniques for Congestion Detection and Avoidance," SIGCOMM '94 Conference on Communications Architectures and Protocols, London, United Kingdom, pp.24-35, Oct, 1994.

[13] V. Jacobson, "4BSD Header Prediction," ACM Computer Communication Review, Apr, 1990.

[14] http : //www-mash.cs.berkeley.edu/ns/.

[15] V. Paxon, S. Floyd, "Wide Area Traffic : The Failure of Poisson Modeling," Proc. Of SIGCOMM '94, pp.257-268, Sep, 1994.

[16] ATM Forum, "ATM Forum Traffic Management Specification, Version 4.0, aftm-0056.000," Apr, 1996.



박 형 우

e-mail : hwpark@hpcnet.ne.kr

1985년 서울시립대학교 전자공학과(학사)

1996년 성균관대학교 전기전자공학과(석사)

2000년 성균관대학교 전기전자및컴퓨터 공학과 박사수료

2001~현재 KISTI 슈퍼컴퓨팅센터 슈퍼컴퓨팅 개발실장/선임연구원

관심분야 : 인터넷 QoS 및 라우팅, 인터넷 망관리, 인터넷 보안

정 진 욱

e-mail : jwchung@songgang.skku.ac.kr

1974년 성균관대학교 전기공학과(학사)

1979년 성균관대학교 전자공학과(석사)

1991년 서울대학교 계산통계학과(박사)

1982년~1985년 한국과학기술 연구소 소장

1981년~1982년 Racal Milgo Co. 객원연구원

1985년~현재 성균관대학교 전기전자 및 컴퓨터 공학부 교수

관심분야 : 컴퓨터 네트워크, 네트워크 관리, 보안