응용 계층 멀티캐스트 성능 향상 기법
(An Application Layer Multicast Performance Enhancement Scheme)

김 성 훈 † 강 경 란 †† 이 동 만 ††† 모 정 훈 †††
(Sunghoon Kim) (Kyungran Kang) (Dongman Lee) (Jeonghoon Mo)

요 약 IP 멀티캐스트가 현재 보편적 네트워 서비스로 제공되지 못하고 있는 현실적인 한계로 인하여, 다중 수신자를 지원하기 위한 대안으로 응용 계층 멀티캐스트 기법들이 주목을 받고 있다. 응용 계층 멀티캐스트는 데이타의 전송이 참가자인 종단 시스템에 의해 이루어진다. 그러나 종단 시스템은 범용 장비이기 때문에 데이타 전달에 최적화되어 있지 않고, 응용 계층 멀티캐스트 세션에 참여하는 종단 시스템들의 컴퓨팅 능력과 네트워크 대역폭이 다양하기 때문에, 종단 시스템의 성능에 따라 처리 지연 시간이 발생하게 되어 종단 간 지연 시간에 영향을 끼치게 된다. 본 논문은 응용 계층 멀티캐스트 세션에 참여하는 종단 시스템에서의 지연 시간을 모델링하고 기존의 응용 계층 멀티캐스트의 성능에 미치는 영향을 분석한다. 나아가, 종단 시스템에서의 지연과 이들의 다양성을 고려하여 메쉬 기반의 응용 계층 멀티캐스트 세션 성능 향상 기법을 제안하고 이에 대한 성능 평가에서 제안하는 기법이 기존의 기법에 비해 종단 간 지연 시간을 감소시키는 것을 보인다.

키워드 : 응용 계층 멀티캐스트, 지연 모델

Abstract Application layer multicast(ALM) has been developed as an alternative to support the multi-receiver applications due to the slow deployment of IP multicast. Unlike routers in IP multicast, end systems participating in a ALM session are not optimized for relaying data and have various processing and network capacity. Therefore, with the increased role of end systems, the delay incurred at the end systems occupy large portion of the end-to-end delay. In this paper, we model the end system delay of ALM sessions and analyze its impact on the performance of the existing ALM schemes. Furthermore, we propose an enhanced scheme, based on the existing mesh-based scheme, which reflects the end system delay. We also evaluate the performance of the proposed scheme by simulation and show that it reduces the end-to-end delay compared with the existing scheme.

Key words : Application Layer Multicast, Delay Model

1. 서 론

네트워크의 보급 확대와 성능 향상으로 네트워크 게임, 오디오/비디오 컨퍼런싱, 멀티미디어 스트리밍 등과 같이 네트워크 상에서 여러 사용자들 간에 데이타 교환이 이루어지는 다대다(many-to-many) 응용 프로그램들이 주목을 받고 있다. 다중 수신자에게로의 데이타 전달을 지원하는 대표적인 기술로 IP 멀티캐스트가 고려되어 왔으나, 실제 상용 인터넷 상에서 보편적인 서비스로서 제공되지 못하고 있다[1]. 이러한 기술적인 현실과 멀티캐스트 서비스에 대한 요구를 고려하여, IP 멀티캐스트에 대한 대안으로 End System Multicast(ESM)[2], NICE[3], TBCP[4] 등과 같은 다양한 응용 계층 멀티캐스트(application layer multicast) 기법들이 제안되어 왔다. 응용 계층 멀티캐스트 기법에서는 각 참가자인 종단 시스템들이 유니캐스트 연결을 이용하여 가상 네트워크를 구성하고, 가상 네트워크 상에 위치한 종단 시스템(end system)들이 '데이타 전송 트리(data delivery tree)'를 설정한다. 데이타 전송 트리에서 단말 노드가 아닌 종단 시스템은 패킷을 수신하면 자신의 해당 응용 프로그램에게 전달할 뿐만 아니라 패킷을 전송 트리의 자식 노드에 해당하는 종단 시스템들에게 전달한다. 이렇게 함으로써, IP 멀티캐스트에 비해 네트워크 상의 중복된 패킷 전달이 많이 발생하지만, 송신자와 개별 수신자 사이의 일대일 유니캐스트 전달에 비해 멀티

† 비 회 원 : LG전자 정보통신사업본부 연구원
s_kim@lgc.co.kr
†† 정 회 원 : 아주대학교 정보및컴퓨터공학부 교수
korykang@ajou.ac.kr
††† 비 회 원 : 한국정보통신대학교 공학부 교수
dlee@icu.ac.kr
jhmo@icu.ac.kr
논문접수 : 2004년 2월 24일
심사완료 : 2004년 8월 13일

캐스팅의 효과를 볼 수 있다.

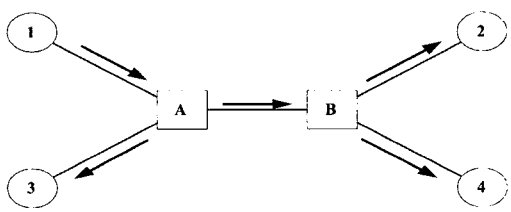
그런데, 종단 시스템은 범용 장비이기 때문에 데이타 전달에 최적화되어 있지 않고, 응용 계층 멀티캐스트 세션에 참여하는 종단 시스템들의 컴퓨팅 능력과 네트워크 대역폭이 다양하다. 그러므로, 종단 시스템이 데이타를 수신하여 이를 분석하고 데이타 전송 트리를 따라 데이타를 전송되는 과정에 있어서 추가적인 지연 시간이 발생한다. 즉, 전송 트리 상의 중간에 위치하게 되는 종단 시스템에 인해 추가되는 '종단 시스템 지연 시간(end system delay)'는 송신자로부터 개별 수신자 사이의 '종단 간 지연 시간(end to end delay)'에 상당한 기여를 할 수 있으며, 참가자들 사이의 동기화(synchronization)를 해침으로써 비디오/오디오 컨퍼런스와 같이 지연에 민감한 응용 프로그램에게는 치명적인 영향을 끼칠 수 있다. 특히, 컴퓨팅 속도가 많이 늦거나 혹은 상당히 작은 네트워크 대역폭을 갖는 종단 시스템이 전송 트리의 중앙에 위치하게 되면, 전송 트리 상에서 이 시스템을 통하여 데이타를 전송받는 종단 시스템의 종단 간 지연 시간이 크게 증가될 수 있다. 그러므로, 비디오 컨퍼런스나 실시간 공유 문서 작업 등과 같이 참가자들 사이의 동기화(synchronization)가 중요한 응용에서는 데이타 전송 트리를 구성하는데 있어 이 같은 종단 시스템에서 발생하는 처리 지연 시간에 대한 고려가 포함되어야 한다. 그러나, 기존의 제안 기법들은 종단 시스템에서 발생하는 지연 시간은 고려하지 않고 네트워크에서 발생하는 지연 시간만을 고려하고 있다.

본 논문에서는 세션의 성능을 '송신자로부터 전송된 데이타 패킷이 모든 수신자에게 전달되는데 걸리는 시간' 즉 '종단 간 지연 시간'이라는 관점에서 측정한다. '성능(throughput)'이 일반적으로 "단위 시간 동안에 한 일의 양"으로 정의되므로, 종단 간 지연 시간을 단축시킬수록 성능을 향상시킬 수 있기 때문이다. 그러므로, 종단 시스템에서 발생하는 지연 시간이 종단 간 지연 시간에 미치는 영향을 분석하고 이를 단축할 수 있는 기법을 개발하는 것이 본 연구의 목적이다. 본 논문에서는 응용 계층 멀티캐스트 세션에 참가하는 종단 시스템을 '도착 큐(incoming queue)'와 '발송 큐(outgoing queue)'로 구성된 시스템으로 모델링하고, 도착 큐와 발송 큐에서 발생하는 지연 시간에 대한 모델을 제시한다. 제안된 모델을 기반으로 개별 큐에서 발생하는 지연 시간과 데이타 전송 트리의 자식 노드의 개수 등을 변화시키면서 종단 시스템 지연 시간에 영향을 끼치는 요인에 대하여 분석한다. 그리고, 개별 참가자 노드에 해당 모델을 적용하였을 때, 세션 참가자의 수의 변화에 따른 기존의 기법인 ESM과 NICE의 종단 간 지연 시간의 변화를 시뮬레이션을 사용하여 관찰한다. 더 나아가, ESM을 기반으로 하여 종단 시스템 지연 시간을 고려하여 가상 네트워을 구성하는 기법을 제안하고 이에 대한 성능 평가를 수행한다. 성능 평가 결과에 따르면, 본 논문에서 제안하는 기법이 ESM에 비해 최대 종단 간 지연 시간을 30% 감소시킨다.

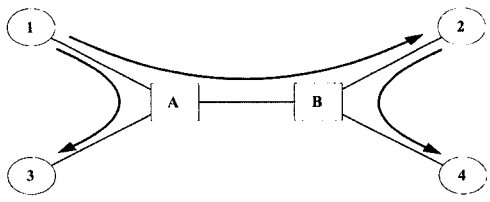
본 논문은 다음과 같이 구성된다. 2장에서는 기존에 제안된 다대다 세션을 위한 응용 계층 멀티캐스트에 대하여 기술한다. 3장에서 종단 시스템 지연 모델을 제안하고 제안된 모델을 이용한 종단 시스템 지연 시간 분석 결과를 기술하고 기존에 제안된 기법들에 대해서도 종단 시스템 지연이 전체 종단 지연 시간에 끼치는 영향을 관찰한다. 4장에서는 종단 시스템 지연 시간을 추가적으로 고려하여 세션의 성능을 향상시킬 수 있는 기법을 제안하고 평가한다. 마지막으로 5장에서 본 연구의 결론과 앞으로의 연구 방향을 제시한다.

2. 응용 계층 멀티캐스트 관련 연구

IP 멀티캐스트 [5]에서는, 그림 1(a)에서 보이듯이, 송신자가 하나의 패킷에 대해 한 번만 송신하면, 네트워상에 존재하는 라우터에 의해 패킷들이 복제되고 라우팅되어 다수의 수신자들에게 전달된다는 장점을 갖는다. 그러나, 기술적인 측면 및 관리적인 측면에서의 문제점으로 인해 상용 인터넷 상에서 범용 서비스로서 사용되지 못하고 있다[1]. 이에 대한 대안으로 고려되고 있는 응용 계층 멀티캐스트에서는, 그림 1(b)에서 보이듯이, 세션 참가자인 개별 종단 시스템들 간의 유니캐스트 연결을 이용하여 패킷을 복제하고 라우팅하여 다수의 수신자들에게 전달한다. 응용 계층 멀티캐스트가 IP 멀티캐스트와 달리 네트웍 상에서 중복된 패킷의 전달을 발



(a) 멀티캐스트에서의 전송



(b) 응용 계층 멀티캐스트에서의 전송

그림 1 IP 멀티캐스트와 응용 계층 멀티캐스트 비교

생시키지만, 네트웍 단계에서 일상적으로 사용 가능한 유니캐스트를 활용하기 때문에 IP 멀티캐스트에 비해 쉽게 도입될 수 있다는 장점을 갖는다.

현재까지 제안된 응용 계층 멀티캐스트 기법들은 주로 일대다 응용, 즉, 서버로부터 데이타를 수신하는 응용 등을 고려하여 개발되었으며[6-10], 본 논문에서 관심을 갖는 다대다 응용을 위해 제안된 기법으로는 ESM[2], NICE[3], TBCP[4] 등이 있다.

TBCP [4]는 대표적인 트리 기반의 응용 계층 멀티캐스트 기법으로서 개별 참가자가 멀티캐스트 그룹에 가입하면서 동시에 데이타 전송 트리를 구성한다. 새로운 참가자는 트리의 루트로부터 자신까지의 거리를 고려하여 전체 참가자들 중 가장 가깝다고 판단되는 일부 이웃 참가자들과의 메시지 교환을 통해서 전송 트리 상에서의 자신의 위치를 결정한다.

ESM [2]은 대표적인 메쉬 기반 기법으로서 기본적으로 참가자의 수가 작은 멀티캐스트 세션에 사용하기 위해 설계되었다. 멀티캐스트 세션에 참여하는 종단 시스템들이 메쉬 구조의 가상 네트워크를 구성한 후, 메쉬 상에서 송신자 기반의 최단 경로 스패닝 트리(shortest path spanning tree)에 해당하는 데이타 전송 트리를 형성한다. 종단 시스템에서 발생하는 지연 시간에 대해서는, 지연이 없거나 혹은 지연 시간이 참가자들마다 모두 동일하다고 가정하고 있다. 각 종단 시스템은 나머지 참가자들에 대해 해당 참가자로의 경로 상의 이웃 노드에 해당하는 종단 시스템과 해당 경로의 비용을 기록한 라우팅 테이블을 유지하고, 주기적으로 라우팅 테이블을 메쉬 상의 이웃 종단 시스템과 교환한다. 그러나, 메쉬를 유지하기 위해 새 참가자가 추가되거나 기존 참가자가 탈퇴할 때마다 기존의 참가자들이 라우팅 테이블을 갱신해야 하고, 참가자의 수가 늘어나게 되면 교환해야 하는 라우팅 테이블의 크기가 비례적으로 커지기 때문에 비교적 확장성이 떨어진다고 할 수 있다[3].

NICE[3]는 멀티캐스트 세션에 참여하는 종단 시스템들 상호 간의 거리를 측정하여 인접한 지역에 위치한 종단 시스템들끼리 클러스터링하고, 각 클러스터의 중앙에 위치하는 종단 시스템을 대표로 선택한다. 이 대표들을 하나의 계층으로 간주하여 새로운 클러스터를 구성한다. 이러한 과정을 반복하여 다계층 구조의 가상 네트워크를 형성하여 응용 계층의 멀티캐스팅을 수행하는 기법이다. 가상 네트워크 상에서 각 종단 시스템은 자신이 속하는 클러스터 내의 종단 시스템만을 인지하므로, 데이타 전송 트리를 위한 가상 네트워크의 유지 관리에 있어서 확장성이 있다고 볼 수 있다. 그런데, 개별 종단 시스템은 자신이 발생시킨 데이타에 대해, 자신이 속하는 모든 계층에서 같은 클러스터에 속한 종단 시스템들에게 데이타를 전송한다. 수신한 데이타에 대해서는, 해당 데이타를 전달해 준 종단 시스템이 속한 계층을 제외한 나머지 자신이 속한 모든 클러스터의 종단 시스템들에게 데이타를 복제해서 전달해야 한다. 따라서, 상위 계층으로 갈수록 속한 클러스터의 수가 증가하므로 데이타를 전송해야 하는 대상이 증가하게 된다. 즉, 상단 계층에 속한 종단 시스템에 트래픽이 집중되어 병목 현상이 발생할 수 있으며, 종단 간 지연 시간을 크게 증가시킬 수 있다.

3. 종단 시스템 지연 모델

3.1 응용 계층 멀티캐스트 세션 성능

일반적으로 '성능(throughput)'을 "단위 시간 동안에 완료한 일의 양"이라고 정의할 때, 응용 계층 멀티캐스트 세션의 성능은 "단위 시간에 수신자들에게 중복되지 않고 전달된 패킷의 수"라고 정의할 수 있다. 따라서, 이 성능은 송신자로부터 개별 수신자로의 종단 간 전달 지연 시간(end-to-end delay)에 반비례하므로, 종단 간 지연 시간을 측정하는 것으로 응용 계층 멀티캐스트 세션의 성능을 간접적으로 측정할 수 있다.

임의의 패킷 p가 송신자 s로부터 종단 시스템 i까지 전달되기 위해서는, 그림 1(b)에서 보이는 바와 같이, s와 i 사이의 전달 경로에 위치한 종단 시스템과 종단 시스템을 연결하는 네트웍 상의 라우터와 회선들을 거쳐야 한다. 그러므로, 전달 패킷 p의 종단 간 지연 시간, 즉, 송신자 s로부터 종단 시스템 i까지 전달되는 데 걸리는 시간은, 수식 (1)과 같이, s와 i의 전달 경로에 위치한 종단 시스템들의 '종단 지연 시간(end system delay)'의 합과 데이타 전송 트리 상의 두 인접 종단 시스템 간의 데이타 전달 지연 시간인 '오버레이 전송 지연 시간(overlay transmission delay)'의 합으로 이루어진다.

$$ED(p;i) = \sum_{n \in N(s,i)} W(n) + \sum_{l \in L(s,i)} PD(l) \quad (1)$$

여기서, $N(s,i)$는 송신자 s에서 종단 시스템 i까지의 데이타 전송 트리 상의 경로에 존재하는 종단 시스템의 집합이고, $W(n)$은 종단 시스템 n에서 발생하는 지연 시간이며, $L(s,i)$는 송신자 s에서 종단 시스템 i까지의 데이타 전송 트리 상의 경로에 존재하는 유니캐스트 연결들의 집합을 의미한다. 따라서, 오버레이 전송 지연 시간에 해당하는 $PD(l)$은 전송 경로에 존재하는 네트웍 링크에서 발생하는 '큐 지연 시간(queueing delay)'와 '링크 전달 지연 시간(link propagation delay)'를 포괄하게 된다. $PD(l)$은 다양한 방법으로 측정이 가능하며, 예를 들어, 유니캐스트 연결 l의 양단에 있는 종단 시스템들 사이에 'ping'을 수행하여 획득할 수 있다.

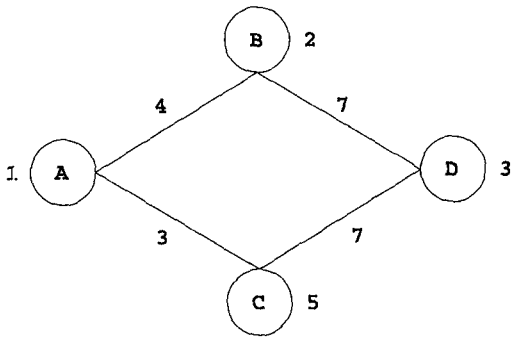


그림 2 메쉬 기반 기법에서의 가상 네트워크 예

그림 2에서 보이는 것과 같이 4개의 노드로 구성된 가상 네트워크를 고려하여 보자. 여기서, 개별 노드는 종단 시스템을 의미하며, 링크는 종단 시스템 사이의 유니캐스트 연결을 의미한다. 링크 상에 표시된 수는 해당 링크의 오버레이 전송 지연 시간이며, 노드 옆으로 표시된 수는 각 종단 시스템이 만들어내는 종단 지연 시간이다.

이러한 가상 네트워크를 기반으로 송신자 A로부터 나머지 종단 시스템으로의 데이타 전송 트리를 구축할 때, 기존의 메쉬 기반 기법에서는 A로부터 D까지의 전송 경로로 C를 경유하는 경로를 선택하게 된다. 오버레이 전송 지연 시간만을 고려하여 경로를 결정하기 때문이다. 그런데, C를 경유하게 되면, N(A,D)와 L(A,D)는 각각 {A, C}과 {(A,C), (C,D)}가 되고, C에서 발생하는 종단 시스템 지연 시간으로 인하여 A와 D 사이의 종단 간 지연 시간은 아래와 같이 계산되어 결과적으로 16이 된다.

$$ED(p;D)=\sum_{n\in N(A,D)}W(n)+\sum_{l\in L(A,D)}PD(l)=(1+5)+(3+7)=16$$

B를 경유하게 되면, N(A,D)와 L(A,D)는 각각 {A, B}과 {(A,B), (B,D)}가 되고, A와 D의 종단 간 지연 시간이 아래와 같이 계산되어 14가 된다.

$$ED(p;D)=\sum_{n\in N(A,D)}W(n)+\sum_{l\in L(A,D)}PD(l)=(1+2)+(4+7)=14$$

즉, C를 경유하는 경우가 B를 경유하는 경우보다 2만큼 종단 간 지연 시간이 증가하게 된다. 본 예에서 보인 바와 같이, 전송 경로 상에 존재하는 종단 시스템 지연이 누적되게 되면 종단 간 지연 시간을 증가시키게 되고 결국 세션 성능을 저하시키게 된다. 바꾸어 생각하면, 수식 (1)에서 정의된 종단 간 지연 시간의 요소들을 단축함으로써 전체 전달 지연 시간이 증가하는 것을 방지할 수 있다. 기존의 응용 계층 멀티캐스트 기법의 연구에서 두 번째 요소에 주력하여 연구를 진행해 온 것에 반하여, 본 논문에서는 두 번째 요소를 크게 증가시키지 않으면서 첫 번째 요소인 종단 시스템에서 발생하는 지연 시간을 단축하는 기법을 제안한다.

3.2 종단 시스템 모델

응용 계층 멀티캐스트 세션에서 종단 시스템은 데이타 패킷의 생성자와 소비자의 역할뿐만 아니라 라우터의 역할까지 수행한다. 실제로 응용 계층 멀티캐스트 세션에 참여하는 종단 시스템은 범용 장비로서 라우터와 같이 데이타 전송에 최적화되어 있지 않으며, 컴퓨팅 능력과 네트워크 대역폭 등의 측면에서 상당한 다양성을 갖는다. 또한, 생성된 데이타 전달 트리를 따라 전송되는 과정에 있어 종단 시스템에서 추가하는 지연 시간들은 - 이후로는 '종단 시스템 지연 시간'이라 칭한다 - 종단 간 지연 시간에 상당한 기여를 할 수 있으며, 비디오/오디오 컨퍼런스와 같은 동기화를 요구하고 지연에 민감한 응용 프로그램에 있어서 치명적인 영향을 끼칠 수 있다.

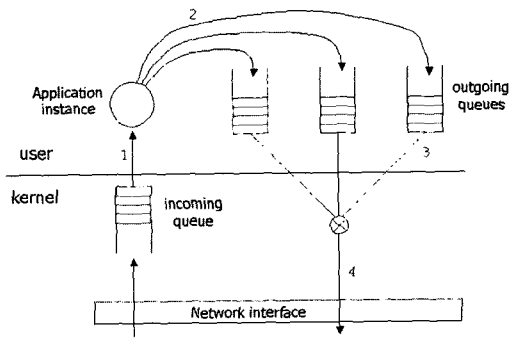


그림 3 종단 시스템 내에서의 패킷 처리 모델

그림 3은 그림 1(b)의 2번 참가자와 같이 데이타 전송 트리에서 전달 기능을 수행해야 하는 종단 시스템에서의 패킷 처리하는 과정을 모델링한 것이다. 종단 시스템은 데이타 전송 트리의 부모 노드로부터 네트워을 통해 패킷을 받으면 '도착 큐(incoming queue)'에 저장한다. 도착 큐에 있는 패킷을 추출하여 헤더를 검사하고, 이를 응용 프로그램에게 전달한 뒤, 이를 자식 노드의 수만큼 존재하는 '발송 큐(outgoing queue)'에 복사한다. 발송 큐에 있는 패킷에 대해 순서대로 데이타 전달 트리의 자식 노드들에게 전달하고, 모든 자식 노드들에게의 전달을 마치면, 자식 노드로부터의 수신 성공 보고를 요구하지 않고, 발송 큐에서 해당 패킷을 삭제한다. 자식 노드에게 전달하는 속도는 네트워크 대역폭이 허용하는 속도에 영향을 받게 되는데, 네트웍의 대역폭을 동적으로 측정하는 기법은 본 논문에서 다루지 않는다. 시스템 커널에서 네트웍 인터페이스 내의 버퍼에 여분의 공간이 없는 오류 상황을 인지하여 버퍼에 공간이

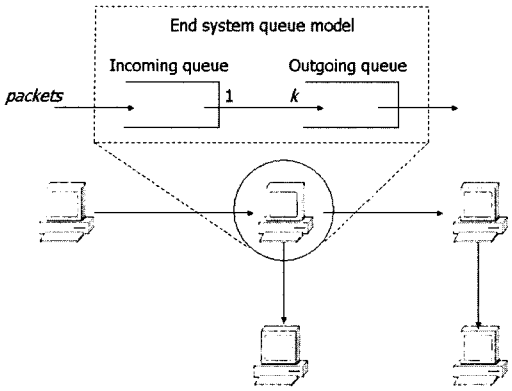


그림 4 단순화된 종단 시스템의 큐 모델

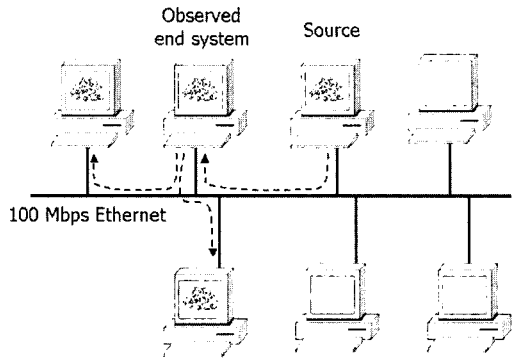


그림 5 도착 큐 지연 시간 실험 환경

생길 때까지 기다리는 것으로 가용 네트웍 대역폭을 인지하는 것으로 가정한다.

위에서 설명한 패킷 처리 모델을 좀 더 단순화하면, 그림 4와 같이 종단 시스템을 두 개의 연속된 큐를 다루는 시스템으로 단순화하여 모델링할 수 있다. 종단 시스템 지연 시간은 수식 (2)와 같이 도착 큐에서의 지연 시간(W_{in})과 발송 큐에서의 지연 시간(W_{out})의 합으로 표현될 수 있다.

$$W = W_{in} + W_{out} \quad (2)$$

앞서 패킷 처리 모델에서 설명한 바와 같이, 도착 큐에 있는 패킷들에 대한 처리 과정에서는 검색, 비교 등과 같이 CPU에 의존하는 과정들이 속하고, 발송 큐에 있는 패킷들에 대해서는 자식 노드의 수만큼 네트웍 인터페이스로 전달하는 과정이 속한다. 즉, 도착 큐 지연 시간은 CPU 성능에 영향을 받고, 발송 큐 지연 시간은 네트웍 대역폭 및 전송 트리 상의 자식 노드의 갯수의 영향을 받을 것이라고 예측할 수 있다.

3.3 도착 큐 지연 시간

그림 3에서 보이는 바와 같이, 도착 큐에 있는 패킷을 처리하는데 걸리는 지연 시간은 다음의 두 가지로 구분된다. 첫째, 네트웍을 통해 수신된 패킷에 대해 운영 체제 커널로부터 응용 프로그램이 인터럽트를 수신하고, 패킷 헤더 검사 및 응용 프로그램에게 패킷을 전달하는데 걸리는 '패킷 처리 지연 시간(packet processing delay)'이다. 둘째, 수신된 패킷을 모든 발송 큐에 복사하는데 걸리는 '패킷 복제 지연 시간(packet replication delay)'이다. 그래서, 도착 큐에서 발생하는 지연 시간(Win)은 수식 (3)과 같이 표현된다.

$$W_{in} = t_r + k \cdot t_q \quad (3)$$

여기서, t_r은 '패킷 처리 지연 시간'을 나타내고, t_q는 하나의 발송 큐에 패킷을 복제하는데 걸리는 시간을 나타낸다. k는 데이타 전송 트리 상의 자식 노드의 수이다. 즉, $k \cdot t_q$는 전체 자식 노드의 수만큼 존재하는 발송 큐 모두에 패킷을 복제하는데 걸리는 시간이 된다.

실제 개인용 컴퓨터를 활용하여 종단 시스템에서 발생하는 도착 큐 처리 지연 시간을 측정하였다. RedHat® 리눅스 버전 8을 운영 체제로 하고 서로 다른 성능을 갖는 시스템에 대하여 도착 큐 지연 시간을 측정하였다. 한 시스템에는 인텔 Pentium®-III 500MHz의 속도를 갖는 CPU를 장착하였고, 다른 한 시스템에는 Pentium®-IV 2GHz의 속도를 가지는 CPU를 장착하였다. 응용 프로그램으로 비디오 컨퍼런싱 프로그램인 vic [11]을 이용하였고, 그림 5에서 보이는 것과 같이, 관찰한 종단 시스템 외에 다른 호스트에서 데이타 소스의 기능을 수행하여 128kbps의 전송률로 데이타를 전송하였다. 관찰한 종단 시스템은 LAN 상에 위치한 임의의 다른 호스트들을 전송 트리의 자식 노드로 간주하여 데이타를 전송하였다. 자식 노드의 수 (k)를 1부터 8까지 두 배씩 증가시키면서 도착 큐 지연 시간을 관찰하였다.

실험 결과는 표 1과 같다. 도착 큐에서 발생하는 패킷 처리 지연 시간(t_r)은 동일한 k에 대해 CPU의 속도에 따라 그 값이 3배 가량 차이를 보이는 것을 관찰할 수 있다. W_{in}도 Pentium®-IV CPU를 가지는 시스템에서의 지연 시간은 1ms가 되지 않는 반면, Pentium®-III의 CPU를 가지는 시스템에서 측정된 지연 시간은 2ms 이상이 된다. 즉, 도착 큐 지연 시간은 CPU의 클락 속도에 영향을 받는다고 할 수 있다. 일반적으로 광역 네트웍 회선의 전달 지연 시간(link propagation delay)가 10ms보다 작은 것을 고려한다면, 관찰된 도착 큐 지연 시간이 전달 지연 시간의 20%를 넘게 된다. 그리고, [12]에서 관찰한 결과에 따르면 전송 트리 상의 라우터 사이의 지연 시간이 1ms 보다 작은 경우가 80%를 넘으므로, 종단 시스템에서의 지연 시간이 연속된 라우터 사이의 지연 시간의 두 배 이상을 차지하는 경우가

표 1 측정된 도착 큐 지연 시간

CPU	k	Delay (ms)		
		t_r	t_q	W_{in}
Pentium® III	1	2.444	0.045	2.290
	2	2.199	0.036	2.192
	4	2.255	0.039	2.369
	8	2.244	0.024	2.434
Pentium® IV	1	0.731	0.015	0.746
	2	0.689	0.012	0.711
	4	0.728	0.013	0.779
	8	0.865	0.013	0.968

80%를 넘게 되어 전체 종단 간 지연 시간에 미치는 영향이 라우터 및 네트워 상에서 발생하는 지연 시간보다 커지게 될 수도 있다.

패킷 처리 지연 시간(t_r)이 전송 트리의 자식 노드의 수 k의 변화에 따라 조금씩 변화하지만 증감의 방향이 k의 변화 방향에 무관한 것을 관찰할 수 있다. 그리고, k가 증가함에 따라 한 번의 패킷 복사를 위한 시간 t_q는 감소하지만 k가 커지므로 전체 패킷 복사에 걸리는 시간($k \cdot t_q$)은 다소 증가한다. 그러나, 결과적으로 계산된 도착 큐 지연 시간(W_{in})을 살펴보면, k의 변화 방향에 무관하게 그 값이 변화하는 것을 관찰할 수 있다. 그러므로, W_{in}이 자식 노드의 수에 영향을 받는다고 말할 수 없다.

정리하면, 응용 계층 멀티캐스트를 위한 종단 시스템에서의 도착 큐 지연 시간은 CPU의 클락 속도에 영향을 받으며, CPU 속도가 낮은 종단 시스템이 전송 트리의 내부 노드로 존재한다면 종단 간 지연 시간 (end-to-end delay)를 증가시켜 전체 세션의 성능을 낮추는 부정적인 영향을 미치게 될 것이라고 할 수 있다.

3.4 발송 큐 지연 시간

발송 큐 지연 시간은 도착 큐 지연 시간이 두 가지 요소로 구성된 것과 마찬가지로 두 가지 요소로 구성된다. 첫째는 전달이 일어나기 전까지 큐 내에서 기다리는 '대기 시간(waiting time)'이고, 둘째는 패킷이 네트워 인터페이스를 통해 네트워 회선 상으로 내 보내지는데 걸리는 '패킷 전달 지연 시간(packet transmission delay)'이다.

미국 AT&T 연구소에서 인터넷의 트래픽을 분석한 결과를 정리한 [13]에 의하면, 부하가 높아질수록 패킷의 도착 분포가 포아송 분포에 근접하므로, 패킷 입력 분포를 포아송 분포로 가정한다. 그리고, 한 패킷을 전달하는데 걸리는 패킷 전달 지연 시간은 네트워크 대역폭이 허용하는 한 거의 일정하기 때문에, 각 종단 시스템의 발송 큐를 M/D/1 큐 모델[14]에 대응시킬 수 있다. 패킷의 길이를 l, 네트워크 대역폭을 b라 하고, 평균적인 패킷 전달을 위한 전달 지연 시간을 S로 표시하면, 수식 (4)와 같이 표현된다.

$$S = \frac{l}{b} \tag{4}$$

M/D/1 큐[14]에서 정의하고 있는 대기 시간 공식을 활용하여 발송 큐에서의 대기 시간에 대한 공식을 유도하면 다음과 같다. 우선, 단위 시간 (초) 당 발송 큐에 도착하는 데이타 패킷의 개수를 λ_t라 하고, 발송 큐의 수는 전송 트리 상의 자식 노드의 수와 같으므로 k라 하면, 발송 큐의 트래픽 밀도(ρ)는 수식 (5)와 같이 표현된다. 이 때, 3.3 장에서 살펴본 바와 같이, W_{in}은 CPU 성능에 따라 좌우될 뿐, 동일한 시스템 내에서는 일정하게 유지되는 값이므로, λ_t는 하나의 종단 시스템에 도착하는 멀티캐스트 세션의 데이타 패킷의 도착률과 같다고 가정해도 무리가 없다.

$$\rho = \lambda_t \cdot k \cdot S = \frac{\lambda_t \cdot k \cdot l}{b} \tag{5}$$

(5)에서 표현된 트래픽 밀도를 사용하여 큐 대기 시간의 공식을 유도하면, 발송 큐의 평균 지연 시간은 수식 (6)과 같이 유도된다.

$$W_t = S + \frac{\rho \cdot S}{2(1-\rho)} = \frac{l}{b} + \frac{\lambda_t \cdot k \cdot l^2}{2b(b - \lambda_t \cdot k \cdot l)} \tag{6}$$

수식 (6)에서 알 수 있듯이, 패킷의 전송 지연은 전송 트리 상의 자식 노드의 수와 평균 패킷의 도착률에 비례하여 증가하는 반면, 네트워크 대역폭과 패킷의 길이에 반비례한다. 또한, 네트워크 대역폭이 감소할수록 전송 트리 상의 자식 노드의 수가 전송 지연에 미치는 영향은 더욱 더 크게 나타난다. 즉, 같은 네트워 대역폭을 갖더라도 자식 노드의 개수가 많은 종단 시스템에서의 발송 큐 지연 시간이 크다.

그림 6은 (6)으로 표현된 발송 큐 지연 시간 공식을 활용하여, 전송 트리의 자식 노드의 수(fanout degree, k), 네트워크 대역폭 (b)이 발송 큐 지연 시간에 미치는

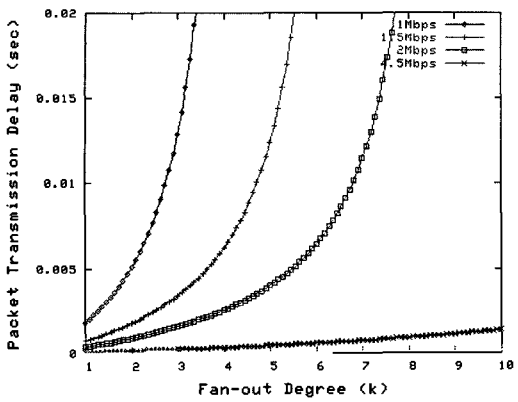


그림 6 전송 트리의 자식 노드의 수(fanout degree)와 네트웍 대역폭이 발송 큐 지연 시간에 미치는 영향

영향을 그래프로 보여준다. 패킷의 길이 l을 1560 바이트로 설정하였으며, 패킷의 도착률 λ_t를 128kbps로 설정하였다. k는 1부터 10까지, 대역폭 b는 1Mbps에서 10Mbps까지 변화시켰다. 이미 수식 (6)에서 예측된 바와 같이, b가 일정할 때 k의 증가에 따라 급격하게 지연 시간이 증가하며, 동일한 k에 대해 b가 증가하면 지연 시간이 급격히 감소하게 된다.

앞서 3.3 장에서 관찰한 표 1과 그림 5에 제시된 값들을 참고하면, 도착 큐 지연 시간은 수 ms에 그치는 것에 비해 발송 큐 지연 시간은 10ms에서 많게는 100ms 이상까지 관측되고 있다. 즉, 종단 시스템 지연 시간은 CPU의 속도보다는 종단 시스템이 갖는 네트워크 대역폭에 영향을 더 받는다고 할 수 있다.

3.5 기존 기법에서의 종단 시스템 지연 시간의 영향 분석

기존의 응용 계층 멀티캐스트 기법인 NICE와 ESM에 종단 시스템의 지연 모델을 적용했을 때의 성능 변화를 관찰하기 위해 네트웍 시뮬레이터 NS-2 [15]를 이용하여 실험을 시행하였다. 사용된 네트워크 위상은 인터넷 위상 생성 도구인 GT-ITM [16]을 이용하여 생성하였다. 네트워크 위상은 transit-stub 모델을 사용하여 만들었고, 전체 500개의 노드가 존재하며, 4 개의 transit 도메인이 있는데, 각 transit 도메인에는 평균 5 개의 transit 노드가 존재한다. 각 transit 노드는 3 개의 stub 도메인을 가지며 각 stub 도메인은 평균 8 개의 종단 노드를 가진다. 또한 모든 링크의 전송 지연은 5ms로 설정하였다.

세션에서 한 참가자만 데이타를 전송하도록 설정하였으며, 128kbps의 속도로 데이타를 전송한다. 개별 패킷에 대해 송신 시각을 표시해서 전송함으로써 개별 수신자까지의 종단 간 지연 시간을 측정할 수 있도록 하였다. 종단 시스템 환경의 다양성을 부여하기 위해서 종단 시스템에 해당하는 노드의 회선 대역폭으로 1Mbps부터 4.5Mbps까지의 값을 동일한 확률로(uniformly) 할당하였고, 도착 큐 지연 시간은 1ms에서 4ms까지의 값을 동일한 확률로 할당하였다. 세션에 참가하는 종단 시스템의 수(group size)를 8부터 64까지 증가시켜 갔으며, 참가자는 stub 도메인에 있는 노드들 중에서 임의로 선택하였다. 동일한 조건의 실험을 10번 이상 반복하여 평균 전송 지연 시간을 측정하였다.

실험 결과가 그림 7에 정리되어 있다. 그림에서 보이는 바와 같이, NICE의 경우 세션의 그룹 크기(group size)가 증가함에 따라 종단 간 지연 시간이 갑작스럽게 증가하게 된다. 이는 그림 8에서 보이는 바와 같이, NICE가 만들어내는 가상 네트워크의 특성 상, 데이타 트래픽과 전송 작업량이 상위 계층에 존재하는 몇몇 리더에게 집중되어 해당 리더의 발송 큐에서의 대기 시간이 크게 증가하기 때문이다. 그림 7에서 가장 높은 계층

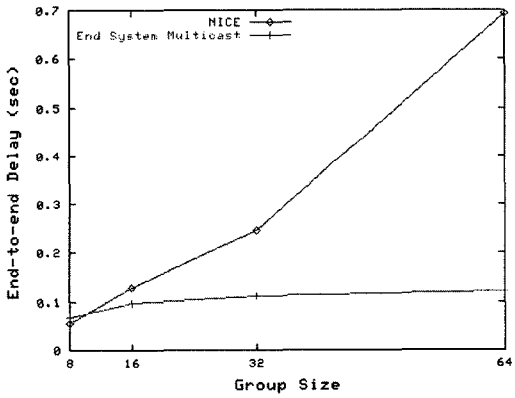


그림 7 End System Multicast와 NICE의 종단 간 지연

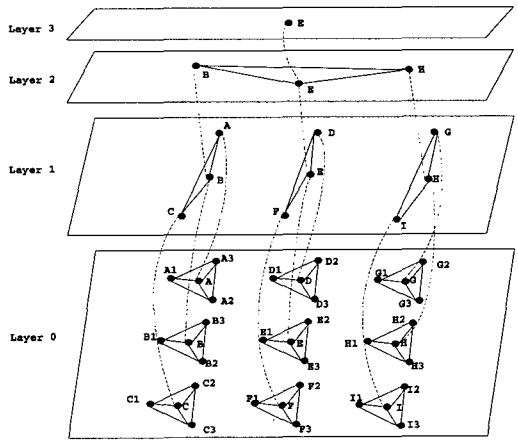


그림 8 NICE에서 만들어내는 계층적 클러스터링 구조

에 속한 종단 시스템 E는 layer 1부터 layer 3에 이르기까지 같은 클러스터에 속한 종단 시스템들에게 패킷을 전달해야 하므로, 최대 7번의 중복 전송이 일어나게 된다. 즉, 전체 30개의 종단 시스템의 25%에 해당하는 종단 시스템에게 데이타를 전달하도록 부담이 집중되므로, 종단 시스템 E에서 발생하는 발송 큐 지연 시간이 커지게 되는 것이다.

반면, ESM의 경우는 종단 간 지연 시간이 세션 참가자 수의 증가에 따라 점진적으로 증가하는 것을 볼 수 있다. 이는 그림 9에서 보이는 바와 같이, 전달 트리의 개별 노드들에게 전달 부담이 분산되기 때문이다.

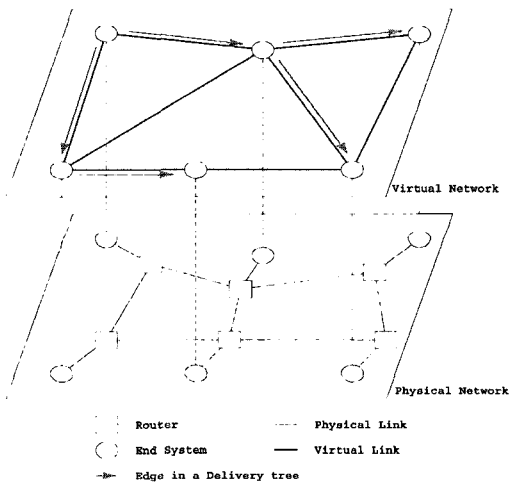


그림 9 ESM에서 구축하는 메쉬 형상의 가상 네트워

그러나, 그림 7에서 세션 참가자 수(group size)가 8일 때와 같이 세션의 참가자 수가 아주 작을 때는, 두 가지 기법 모두 자식 노드의 수가 크게 증가하지 않기 때문에 두 기법의 종단 간 지연 시간이 크게 차이를 보이지 않는다.

4. 세션 성능 향상 기법

4.1 메쉬 기반 성능 향상 기법

3장의 실험 결과에서 볼 수 있듯이, NICE와 같은 다계층 클러스터링 구조는 종단 시스템에서 발생하는 지연 시간으로 인해 세션의 참가자 수가 증가하게 되면, 전체 세션의 성능을 크게 저하시킬 수 있다. 그러나, 메쉬 구조를 기반으로 하는 ESM은 비교적 참가자의 수에 대해 지연 시간의 증가가 크지 않음을 관찰할 수 있었다. 하지만, ESM 역시 데이타 전송 트리를 구성하는데 있어 종단 시스템의 지연 시간을 고려하지 않았기 때문에, 3.1장의 예제에서 설명한 것과 같은 잘못된 경로를 선택하게 될 가능성을 갖고 있다. 그래서, 이러한 문제점을 해결하기 위해, 본 장에서는 ESM을 기반으로 하여 종단 시스템 지연 시간을 고려하는 전송 트리 구성 기법을 제안한다.

제안하는 기법은 전송 트리 상의 모든 경유 노드들의 종단 시스템 지연 시간을 반영하여 종단 간 지연 시간을 측정하고 이를 기반으로 데이타 전송 트리를 구성한다. 그래서, 여러 가지 경로 중에서 가장 낮은 종단 간 지연 시간을 만들어내는 경로를 선택함으로 세션 성능의 향상을 도모한다. 이를 위해서 각 종단 시스템들이 주기적으로 자신이 만들어내는 종단 시스템 지연 시간의 예상치를 계산하여 메쉬 상의 이웃 종단 시스템들에게 알린다.

비디오 화상 회의와 같은 고정적인 데이타 전송률을 보이는 응용을 고려하여, 3.4 장에서 기술한 발송 큐 지연 시간 모델에서 패킷의 입력 속도가 일정하다고 가정하자. 그러면, 한 종단 시스템이 i번째 자식 노드에게 전송하기까지 걸리는 종단 시스템 지연 시간의 예상치는 수식 (7)과 같이 계산될 수 있다.

$$W(i) = W_{in}(i) + W_{out}(i) = t_r + \frac{l}{b} \cdot i \tag{7}$$

여기서, t_r은, 3장에서 설명한 바와 같이, 패킷 처리에 걸리는 시간을 나타내고, l은 평균 데이타 패킷의 길이를, l/b는 패킷 전달 지연 시간을 나타낸다. 3장에서 설명한 바와 같이 $W_{in}(i)$는 t_r과 $k \cdot t_q$로 이루어지는데, 표 1에서 보이는 바와 같이 $k \cdot t_q$가 t_r에 비해 크게 작으므로 (7)에서는 고려하지 않는다. 그리고, 3장에서 설명한 바와 같이 t_r이 자식 노드의 수에 무관하고 개별 종단 시스템의 CPU 성능에 좌우되므로, 동일한 종단 시스템에서는 임의의 자식 노드에 대해서도 일정한 값으로 유지된다. 하지만, 즉, i번째 발송 큐에 있는 패킷이 네트워 인터페이스를 통해 전송되기 위해서는 (i-1)번째 발송 큐의 패킷까지 전송이 완료되고 나서 해당 패킷이 전송되므로, 발송 큐 지연 시간 $W_{out}(i)$는 전자에 걸리는 시간 $i \cdot l/b$과 후자에 걸리는 시간 l/b의 합인 $i \cdot l/b$로 나타내어진다.

그림 10은 이렇게 계산된 종단 시스템 지연 시간을 이웃한 시스템과 교환하는 알고리즘을 보이고 있다. 기존에 ESM에서 이웃 시스템과 라우팅 테이블을 주기적으로 교환하는데, 이 라우팅 테이블에 종단 시스템에 대한 정보를 포함하여 전송함으로써 자신의 종단 시스템 지연 시간을 이웃에게 알린다(*tx_rt_exchange*()). 그리고, 라우팅 테이블에서 그리고, 이웃으로부터 라우팅 테이블을 수신하면, 이에 동반하여 전달된 종단 지연 시간을 추출하여 해당 이웃 시스템을 경유하는 경로에 대해

```
//Sender
tx_rt_exchange()
{
// neighbor_list is the list of the neighbor members
for (each neighbor i in neighbor_list)
{
// Win is the incoming queue delay
// Wout[ i] is the outgoing queue delay required until completing packet
// transmission to the i-th child end system
      msg.End_System_Delay = Win + Wout[ i] ;
// copy the routing table into the message
      msg.Routing_Table = rt;
      send_mesage(msg, i);
      }
}

//Receiver
rx_rt_exchange(msg)
{
// PD is the propagation delay measured by ping
nbr_cost=PD(msg.sender)+msg.End_System_Delay;
nbr_rt=msg.Routing_Table;
// compare the current path with the path passing by msg.sender
for each session member j
{
      if (rt[ j] .cost > nbrcost+nbr_rt[ j] .cost)
            update next hop for j to msg.sender;
      }
}
```

그림 10 라우팅 테이블 정보 교환 모듈

표 2 종단 시스템 지연 시간을 고려하지 않은 경우, D의 라우팅 테이블

Destination	Next hop	Cost
A	C	10
B	B	7
C	C	7
D	D	0

표 3 종단 시스템 지연 시간을 고려한 경우, D의 라우팅 테이블

Destination	Next hop	Cost
A	B	14
B	B	7
C	C	7
D	D	0

서 비용을 계산한다. 해당 이웃 시스템을 경유하는 경로의 비용이 더 낮다면 경로를 수정한다(*rx_rt_exchange*()). 여기서, 추가로 전달되는 정보가 계산된 종단 지연 시간 뿐이므로 전체 라우팅 테이블 교환 시에 발생하는 트래픽의 증가에 미치는 영향은 미비하다. 그리고, 라우팅 테이블을 갱신하는 것은 기존의 ESM에서도 라우팅 테이블 교환 후에 시행되었던 것이고, 간헐적으로 발생하는 처리이므로 개별 종단 시스템에서 야기되는 처리 부담 또한 크지 않다.

그림 2의 경우를 예로 하여 제안하는 기법이 동작하는 과정을 간단하게 설명하면 다음과 같다. 종단 시스템 지연 시간을 고려하지 않으면, D는 표 2와 같은 라우팅 테이블을 갖게 되고, A로 가는 경로로 C를 선택한다. 하지만, 그림 9에서 기술하고 있는 알고리즘에 따라 C와 B로부터 종단 지연 시간과 라우팅 테이블을 받아서 C에서 발생하는 종단 지연 시간과 B의 종단 지연 시간을 포함하여 비용을 새로 계산하게 되면, C를 경유하여 A로 가는 경우 비용이 16이 되고, B를 경유하면 비용이 14가 되므로, 표 3과 같이 라우팅 테이블이 변경되게 된다.

4.2 성능 평가

네트웍 시뮬레이터를 사용하여 기존의 ESM과 본 논문에서 제안하는 기법의 성능을 비교한다. 그림 7에서 보이는 바와 같이 NICE의 종단 간 지연 시간이 ESM에 비해 최대 10 배 가량 크게 나타나기 때문에 본 성능 평가에서는 NICE를 고려하지 않는다. 시뮬레이션 환경은 표 4에서 보이는 바와 같이 3장에서의 실험 환경과 동일하며, 동일한 환경에서 10회 이상 반복하여 수집된 결과에 대해 평균값과 최대값을 추출한다.

표 4 성능 평가 환경

환경 변수	값	비고
전체 노드 수	500	transit stub 모델 - 4 개의 transit 도메인 -한 transit 도메인 당 5개의 transit 노드 -한 개의 transit 노드 당 3개의 stub 도메인 -한 개의 stub 도메인 당 8개의 종단 노드
세션 참가 노드 수 (group size)	8, 16, 32, 64, 128	종단 노드들 중에서 임의로 선택
데이타 전송률	128kbps	송신자의 데이타 전송 속도
네트워 회선 대역폭	1Mbps, 10Mbps	두 값 중 임의로 선택하여 설정
회선 전송 지연 시간	5ms	모든 회선에 동일하게 적용
도착 큐 지연 시간	1ms, 4ms	두 값 중 임의로 선택

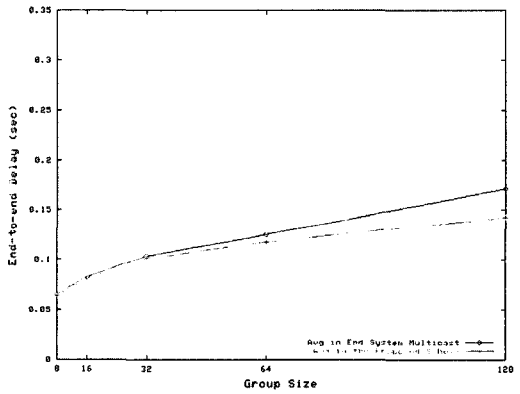


그림 11 기존의 ESM과 제안하는 기법의 평균 종단 간 지연 시간 비교

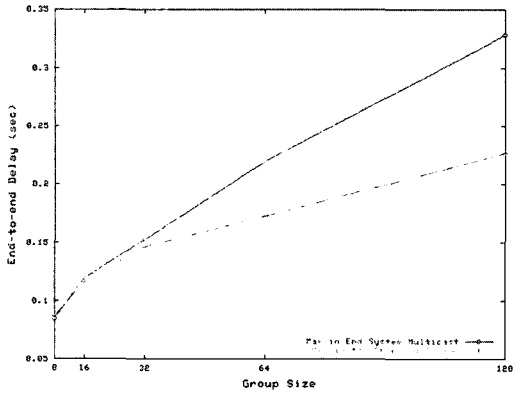


그림 12 기존의 ESM과 제안하는 기법의 최대 종단 간 지연 시간 비교

그림 11과 12는 각각 ESM과 제안된 기법에서 측정된 종단 간 지연 시간의 평균과 최대값의 평균을 보여주고 있다. 그림 11과 12에서 참가자의 수가 증가함에 따라 제안된 기법의 종단 간 지연 시간과 ESM의 종단 간 지연 시간의 차이가 점점 더 증가하는 것을 볼 수 있다. 그 이유는 세션의 참가자 수가 많아질수록 자식 노드의 수가 증가하게 될 확률이 커지므로 거의 비슷한 값의 오버레이 전송 지연 시간을 가지면서 서로 다른 종단 시스템 지연을 가지게 되는 경로들이 생길 확률이 더욱 더 커지게 되기 때문이다. 본 논문에서 제안하는 기법을 사용하면, 이러한 여러 경로 중 가장 혹은 보다 낮은 종단 간 지연 시간을 가지는 경로를 선택하게 되기 때문에, 종단 간 지연 시간을 고려하지 않은 기존의 ESM보다 낮은 종단 간 지연 시간을 보이게 된다.

그리고, 더 나아가, 그림 12에서 보이는 바와 같이, 세션의 참가자 수가 128인 경우, 제안하는 기법의 종단 간 지연 시간 최대값이 ESM의 최대값에 비해 30% 가량 단축되었다. 이는 거의 비슷한 오버레이 전송 지연 시간을 가지는 여러 경로 중 보다 작은 종단 간 지연을 보이는 경로를 종단 시스템에서 선택할 수 있도록 하였기 때문이다.

5. 결론 및 향후 연구 계획

응용 계층 멀티캐스트 기법은, IP 멀티캐스트와는 달리 세션에 참가하는 종단 시스템에서 데이타를 복제하여 이웃 시스템에게 전달하는 식으로 멀티캐스팅이 이루어진다. 그러므로, 기존의 네트워크 자원의 사용에서의 효율성뿐만 아니라 멀티캐스트 라우팅에 참여하는 종단 시스템들이 데이타 전달에 최적화되어 있지 않다는 특징을 고려하여 볼 때, 데이타 전달을 위한 처리 지연 또한 성능에 영향을 미치는 요소가 된다.

본 논문에서는 응용 계층 멀티캐스트 기법에서 데이타 전달을 수행하는 종단 시스템에서의 지연을 M/D/1 큐 모델에 근거하여 모델링하였으며, 종단 시스템에서의 지연이 종단 간 지연 시간에 미치는 영향을 고찰하였다. 또한, 제안된 모델을 기반으로 종단 시스템에서의 지연 요소와 기존 응용 계층 멀티캐스트 기법인 End Sys-

tem Multicast(ESM)과 NICE의 성능 분석을 수행하였다. ESM에 비해 확장성이 있는 가상 네트워크 구조를 구성한다고 평가되는 NICE의 경우에도, 참가자의 수가 증가할수록 상위 계층의 종단 시스템에서 전달해야 하는 이웃 시스템의 수가 급격하게 증가하게 되어 전체 세션에서 종단 시스템간의 전송 지연을 증가시키는 것을 관찰할 수 있었다. ESM은 각 종단 시스템에서의 최대 스트레스를 제한함으로써 NICE와 같은 급격한 종단 시스템 지연 시간의 증가를 유발하지 않았지만 점진적으로 증가하는 것을 관찰할 수 있었다. 이러한 실험 결과를 기반으로 하여, ESM을 기반으로 종단 시스템에서의 지연 시간을 반영하도록 확장한 메쉬 기반 응용 계층 멀티캐스트 기법을 제안하였으며, 종단 간 지연 시간을 단축함으로써 세션의 성능을 향상시킬 수 있게 하였다. 시뮬레이션을 통한 성능 평가에서, 제안하는 기법이 기존의 ESM에 비해 그룹의 크기가 128일 경우 최고 30%의 성능 개선이 이루어짐을 관찰할 수 있었다.

본 논문의 주된 기여점은 응용 계층 멀티캐스트에서 종단 시스템에서 발생하는 지연 시간이 전체 세션의 성능에 영향이 있음으로 보이고, 이러한 지연 시간을 반영하여 응용 계층 멀티캐스트 기법을 개발할 수 있는 방법을 제시했다는 것이다. 그리고, 앞으로 실제 네트웍 및 시스템을 활용하여 시뮬레이션을 통해 관찰된 결과들을 확인하도록 할 것이며, 보다 다양한 응용의 경우를 지원하기 위해 가변 데이타 전송률을 보이는 실제 응용 계층 멀티캐스트 세션의 수행 결과 데이타 등을 활용하여 모델의 타당성 및 실용성을 증가시키도록 할 것이다.

참 고 문 헌

[1] C. Diot, B.N. Levine, B. Lyles, H. Kassem, and D. Balensiefe, "Deployment issues for the IP multicast service and architecture," IEEE Network, Vol. 14, No. 1, pp. 78-88, Jan. 2000.

[2] Y. Chu, S. G. Rao, and H. Zhang, "A Case for End System Multicast," Proc. of ACM SIGMETRICS'00, pp.1-12, 2000.

[3] S. Banerjee, B. Bhattacharjee, and C. Kommareddy, "Scalable Application Layer Multicast," Proc. of ACM SIGCOMM'02, pp. 205-217, 2002.

[4] L. Mathy, R. Canonico, and D. Hutchison, "An Overlay Tree Building Control Protocol," Proc. of NGC 2001, 2001.

[5] S. Deering, Host Extensions for IP Multicasting, Internet RFC 1112, Aug. 1989.

[6] Yatin Chawathe, Steven McCanne, and Eric A. Brewer, "RMX:Reliable Multicast for Heterogeneous Networks," Proc. of IEEE INFOCOM'00, pp. 795-804, 2000.

[7] John Jannotti, David K. Gifford, Kirk L. Johnson, M. Frans Kaashoek, and James W. O'Toole, Jr., "Overcast: Reliable Multicasting with an Overlay Network," Proc. of USENIX Symposium on Operating Systems Design and Implementation, 2000.

[8] Beichuan Zhang, Sugih Jamin, and Lixia Zhang, "Host Multicast: A Framework for Delivering Multicast To End Users," Proc. of IEEE INFOCOM'02, pp.1366-1375, 2002.

[9] Jorg Liebeherr, Michael Nahas, and Weisheng Si, "Application-Layer Multicasting With Delaunay Triangulation Overlays," IEEE Journal on Selected Areas in Communications, Vol. 20, No. 8, pp. 1472-1488, 2002.

[10] Miguel Castro, Peter Druschel, Anne-Marie Kermarrec, Antony Rowstron, "Scribe: A large-scale and decentralized application-level multicast infrastructure," IEEE Journal on Selected Areas in Communications, Vol. 20, No. 8, pp. 1489-1499, 2002.

[11] S. McCanne and V. Jacobson, "vic: a flexible framework for packet video," Proc. of ACM Multimedia, pp. 511-522, 1995.

[12] Sonia Fahmy and Minseok Kwon, "Characterizing Overlay Multicast Networks," Proc. of IEEE ICNP 2003, pp. 61-70, 2003.

[13] C. Holmes, D. Denison, M. Hansen, B. Yu, and B. Mallick, "Internet Traffic Tends Toward Poisson and Independent as the Load Increases," Nonlinear Estimation and Classification, Springer, New York, 2002.

[14] D. Gross and C. M. Harris, Fundamentals of Queueing Theory, 3rd edition, John Wiley & Sons, Inc., pp 116-122, 1998.

[15] NS-2, Network Simulator 2, http://www.isi.edu/nsnam/ns/

[16] K. Calvert, M.B. Doar, and E.W. Zegura, "Modeling Internet Topology," IEEE Communications, Vol. 35, No. 6, pp. 160-163, 1997.(tools are available at http://www.cc.gatech.edu/projects/gtitm/)

김 성 훈

1999년 경북대학교 컴퓨터공학 학사 2004년 ICU 석사. 현재 LG전자 정보통신사업본부 단말연구소 연구원. 관심분야는 멀티캐스트, Wireless communication

강 경 란

1992년 2월 서울대학교 계산통계학과 학사. 1994년 2월 한국과학기술원 석사. 1999년 2월 한국과학기술원 박사. 1999년~2000년 한국전자통신연구원 선임연구원 2000년~2002년 (주)디지탈웨이브 책임연구원. 2002년~2004년 2월 한국정보통신대학원대학교 연구조교수. 2004년 3월~현재 아주대학교 정보및컴퓨터공학부 조교수. 관심분야는 multicast, overlay network, mobility

이 동 만

1982년 서울대학교 컴퓨터공학 학사 1984년 한국과학기술원 전산학 석사 1987년 한국과학기술원 전산학 박사 1987년~1988년 한국과학기술원 박사후 과정. 1988년~1997년 Hewlett-Packard 책임연구원. 1997년~2004년 3월 한국정보통신대학교 부교수. 2004년 4월~현재 한국정보통신대학교 교수. 관심분야는 multicast protocol, group communication, distributed virtual environment, pervasive computing

모 정 훈

학사와 석사는 서울대학교 산업공학과에서 취득함. 1999년에 University of California, Berkeley에서 "Internet Congestion Control"에 관한 연구로 박사 학위 취득함. 2003년 8월부터 한국정보통신대학교 공학부 조교수로 재직 중임 관심분야는 data communication, congestion control, network protocol, wireless ad-hoc networks, cdma/ofdm 등