파일전송의 성능향상을 위한 다중 가상소스 응용계층 멀티캐스트

(Overlay Multicast for File Distribution using Virtual Sources)

이 수 전 † 이 동 만 †† 강 경 란 †††
(Soojeon Lee) (Dongman Lee) (Kyungran Kang)

요 약 응용 계층 멀티캐스트에서 일반적으로 트리를 구성하여 데이타를 전송하는데, 이 경우 트리 상에서 하위(descendants) 노드들이 체감하는 처리율(throughput)은 상위(ancestor) 노드들의 성능에 의해 좌우된다. 그리고, 상위 노드에서 오류가 하위 노드 전체에 영향을 끼친다는 단점을 갖는다. 본 문서에서는 이러한 트리 구조의 약점들을 보완하는 방법을 제안한다. 전송 트리 상의 부모 노드 외의 다수의 가상 소스를 설정하고, 이들로부터 데이타를 수신함으로써 단위 시간 수신량을 늘리며 부모 노드의 오류를 피해갈 수 있는 방안을 마련한다. 그리고, 다수의 가상 소스를 이용할 경우 발생할 수 있는 동일 데이타의 중복 수신 문제를 피할 수 있도록 가상 소스 선택 알고리즘을 제시한다. 응용 계층 멀티캐스트 시험 개발 환경인 MACEDON을 이용하여 시험 구현하였고, 이를 전세계적인 오버레이 네트워크 PlanetLab에 적용하여 성능 평가를 시행하였다. 성능 평가 결과, 기존의 다수 전달자를 제안하는 기법인 Bullet에 비하여 20% 성능을 향상시키면서 동시에 중복 수신되는 데이타 및 제어에 필요한 메시지 비용을 90% 이상 감소시키는 효과를 볼 수 있었다.

키워드 : 다대다 멀티캐스트, 응용 계층 네트워크

Abstract Algorithms for application-level multicast often use trees to deliver data from the source to the multiple receivers. With the tree structure, the throughput experienced by the descendant nodes will be determined by the performance of the slowest ancestor node. Furthermore, the failure of an ancestor node results in the suspension of the session of all the descendant nodes. This paper focuses on the transmission of data using multiple virtual forwarders, and suggests a scheme to overcome the drawbacks of the plain tree-based application layer multicast schemes. The proposed scheme elects multiple forwarders other than the parent node of the delivery tree. A receiver receives data from the multiple forwarders as well as the parent node and it can increase the amount of receiving data per time unit. The multiple forwarder helps a receiver to reduce the impact of the failure of an ancestor node. The proposed scheme suggests the forwarder selection algorithm to avoid the receipt of duplicate packets. We implemented the proposed scheme using MACEDON which provides a development environment for application layer multicast. We compared the proposed scheme with Bullet by applying the implementation in PlanetLab which is a global overlay network. The evaluation results show that the proposed scheme enhanced the throughput by 20 % and reduced the control overhead over 90 % compared with Bullet.

Key words : application layer multicast, overlay networks

† 정 회 원 : 한국전자통신연구원 한국전자통신연구원 연구원
soojeonlee@etri.re.kr
†† 정 회 원 : 정보통신대학교 공학부 교수
dlee@icu.ac.kr
††† 정 회 원 : 아주대학교 정보및컴퓨터공학부 교수
korykang@ajou.ac.kr
논문접수 : 2005년 7월 9일
심사완료 : 2006년 4월 11일

1. 서 론

일대다 스트리밍, 일대다 파일 전송 등의 응용에서 송신자의 데이타 전송 부담 및 네트워크 대역폭 사용량을 줄이기 위해 멀티캐스트를 사용한다. IP 멀티캐스트[1]가 가장 효율적으로 네트워크 자원을 활용하여 멀티캐스트를 지원할 수 있는 방법론으로 여겨지고 있지만, 오

랜 기간의 노력에도 불구하고 여러 가지 장애 요소로 인하여 실제 상용 서비스로는 널리 사용되지 않고 있다[2]. 따라서, 이에 대한 대안으로 응용 계층에서 멀티캐스트를 구현하려는 연구가 활발히 진행되고 있다. IP 멀티캐스트에 비해 전송 부담이나 네트워크 대역폭 사용량은 많지만, IP 멀티캐스트에 비하여 제작 및 적용이 용이하다는 장점을 갖기 때문에 응용 계층 멀티캐스트에 대한 관심이 높다[3].

응용 계층 멀티캐스트 방식에서도, IP 멀티캐스트 방식과 유사하게 데이타 전송을 위하여 소스와 전체 수신자들 사이에 소스가 루트가 되는 스패닝 트리를 구성하는 것이 일반적이다. 그런데, 이러한 트리 구조는 크게 두 가지 약점을 갖는다. 첫째, 트리의 중간 노드 중 네트웍 환경이나 컴퓨팅 성능(performance)이 안 좋은 노드에 의해 그 하위 노드들이 체감하는 처리율(throughput)이 떨어질 수 있다. 둘째, 각 수신자는 단 하나의 부모 노드로부터 데이타를 받아야 하기 때문에 이 부모 노드의 오류(failure)는 자식 노드들에게 데이타 전송 실패, 지연 등 직접적인 영향을 미치게 된다. 이를 극복하기 위해 새로운 부모 노드를 선정하는 데에도 시간이 오래 걸릴 수 있어 전체 데이타를 수신하는데 실패하거나 수신을 완료하는데 걸리는 시간이 길어지게 된다. Narada[4] 등에서는 전송 트리를 메쉬(mesh) 기반으로 하여 갑작스러운 부모 노드의 오류에 보다 유연하게 대처할 수 있지만, 전송 트리를 기반으로 데이타가 전달되기 때문에 앞서 언급한 첫번째 문제점에 대해서는 취약하다. 즉, "자식 노드의 수신 처리율은 부모 노드의 처리율보다 작거나 같다"는 트리 구조의 약점을 여전히 갖게 된다.

이러한 문제점을 해결하기 위해, 전송 트리를 구성하되 부모 노드 외의 다른 노드들로부터도 데이타를 수신하는 '다수 전송달자 기법(multiple forwarder approach)' 응용 계층 멀티캐스트들이 제안되었다. Bullet[5], informed delivery[6], SplitStream[7], 이중 멀티캐스트 메쉬 기법[8] 등이 이러한 기법에 속한다. 이러한 기법들은 다수의 수신자를 대상으로 하는 파일 전송 서비스를 목표 서비스로 하고 있다. 기본적으로 송신자가 데이타를 Forward Error Correction(FEC) 방식으로 인코딩하여 보내기 때문에, 일정한 양만큼의 데이타를 수신하면 원래 데이타를 복원할 수 있으므로 데이타의 순서 번호가 중요하지 않다. 그러므로, 단위 시간 내에 가능한 한 다수의 전달자로부터 중복되지 않는 데이타를 수신하는 것이 가장 중요한 고려 사항이다. 여기서, 데이타의 중복을 피할 수 있도록 전달자를 선택하는 것이 중요하고, 이를 위해서는 각 구성원들의 수신 상태 정보를 확인하는 작업이 필요하다. 이때, 기존의 트리 기반의 전송 기법에 비해 추가적인 제어 부담이 발생하게 된다. P2P 파일 전송 기법에서도 단일 피어(peer)로부터 혹은 소스로부터 데이타를 획득하는 것이 아니고, 파일을 일정한 크기의 조각으로 나누고, 각 조각을 서로 다른 피어에게 요청해서 수신하는 기법들이 개발되고 있다. BitTorrent[9]가 그 대표적인 예라 할 수 있다. BitTorrent는 250KB 단위로 데이타를 분할하고, 각 피어가 자신이 갖고 있는 조각에 대한 정보를 tracker에게 등록한다. 각 피어는 tracker를 통해서 자신이 필요로 하는 조각을 가진 피어를 찾을 수 있다. 이러한 방법은 실제 인터넷 상에서 사용되고 있는 서비스라는 점에서 현실적인 기술이라 평가할 수 있지만, 조각의 크기가 크고, 개별 조각을 다 수신한 후에야 다른 피어에게 해당 조각을 전송할 수 있으므로, 실제 타 피어를 통해서 획득할 수 있는 데이타의 양은 크지 않다.

본 논문에서는 기존의 다수 전달자 기반의 응용 계층 멀티캐스트 기법에서 제기된 제어 부담을 줄이면서 데이타 수신 시간을 줄일 수 있는 기법을 제안한다. 본 논문에서 제시하는 기법의 주된 가정 중의 하나는, 송신자가 어떤 수신자들이 자신의 데이타를 획득하는지 파악해야 한다는 것이다. 즉, 수신자는 송신자에게 데이타 수신 여부를 등록하는 과정이 필요하고, 송신자는 수신자의 초기 수신 상태 정보를 획득할 수 있다는 것이다. 그래서, 본 논문에서 제안하는 기법에서는 개별 수신자가 자신의 처음 수신 패킷 번호 정보를 송신자에게 등록하고 타 수신자에 대한 정보를 송신자로부터 획득하도록 한다. 이렇게 함으로써 타 수신자에 대한 정보를 획득하기 위해 수신자들끼리 주고 받아야 하는 제어 메시지들을 크게 줄일 수 있다. 또 한 가지 중요한 가정은, 수신자들이 서로 다른 시점에 송신자의 데이타를 수신하기 위해 세션에 가입할 것이며, 따라서 개별 수신자가 세션에 가입해서 처음 수신하는 패킷의 순서 번호는 서로 다르다는 것이다. 그러므로, 각 수신자가 자신이 가입한 이후 다음 수신자가 가입하기 전까지 전송 트리 상의 부모 노드로부터 수신한 데이타 패킷들은 타 수신자들과 구별될 수 있고, 이를 타 수신자들에게 전달한다면 부모 노드 외의 추가적인 전달자로부터 데이타 패킷을 수신한다고 해도 데이타의 중복을 피할 수 있다. 그래서, 본 논문에서 제안하는 기법에서는, 부모 노드 외에 추가적으로 데이타를 전송해 줄 전달자를 '가상 소스'라고 칭하며, 해당 수신자보다 먼저 세션에 가입하여 데이타를 수신하고 있는 수신자들 중에 가상 소스를 선택하도록 한다. 제안하는 기법에 대한 성능 평가를 PlanetLab[10] 상에서 수행하였으며, 기존의 다수 전송달자 응용 계층 멀티캐스트 기법인 Bullet과 성능을 비교하였다. 50개의 수신자들에 대해서 실험을 수행하였으

며, 성능 시험 결과 Bullet에 비해 데이타 수신에 걸리는 시간은 20% 가량 감소시켰으며, 제어에 필요한 비용은 90% 이상 감소시키는 효과를 볼 수 있었다.

본 논문은 다음과 같이 구성되어 있다. 2장에서는 파일 전송을 위해 설계된 기존의 응용 계층 멀티캐스트 기법을 단일 전달자 방식과 다중 전달자 방식으로 나누어 분석하며, 3장에서는 제안하는 기법의 설계 고려 사항을 기술하고, 4장에서는 제안하는 기법을 자세하게 기술한다. 5장에서는 성능 평가 결과를 제시할 것이며, 6장에서 결론과 추후 연구 계획을 제시하는 것으로 논문을 마무리한다.

2. 관련 연구

여러 수신자들에게 파일을 전송하면서 수신자들의 협력에 의해 파일의 성능수신 성능을 향상시키기 위한 기술들은 응용 계층 멀티캐스트 외에 P2P에서도 활발하게 진행되어 왔다. 기존의 응용 계층 멀티캐스트와 P2P의 차이점은 P2P는 송신자와 수신자의 관계로 일대일 데이타 송수신을 하는 반면에, 응용 계층 멀티캐스트에서는 각 수신자가 전송 트리의 부모 노드로부터 수신한 데이타를, 라우터가 데이타를 전달하듯, 자식 노드에게 전달해 주는 기능을 수행했다는 점이다. 즉, 응용 계층 멀티캐스트의 수신자는 단순 전달자라는 측면에서 피동적인 반면에 P2P의 수신자는 끊임없이 자신이 필요로하는 패킷의 소유자를 찾는다는 측면에서 능동적이라 차이가 있다. 그런데, 파일 전송의 성능을 향상시키기 위한 기법을 개발하는 과정에서 응용 계층 멀티캐스트의 수신자들이 단순한 전달 기능 외에 자신이 확보한 데이타를 추가적으로 자식 노드 외의 타 노드에게 전달하고 수신한다는 점에서 P2P의 협력적인 파일 송수신 기법과 상당히 유사한 기술로 발전하게 되었다.

다수 전달자 기법의 응용 계층 멀티캐스트로는 Bullet[5], Informed Delivery[6], SplitStream[7], 이중 멀티캐스트 메시 기법[8] 등을 대표적인 예로 들 수 있다. Bullet과 SplitStream은 기본적으로 소스가 데이타를 FEC 특히 Digital Fountain 기법[11]에 의해 인코딩하는 것을 가정한다. 기존의 단순한 전송 트리 외에 타 노드로부터 데이타를 수신하기 위한 TCP 연결을 만들게 되므로 전체적으로 메쉬 구조의 오버레이 네트워크가 구성된다. 이 때, 중복된 데이타 수신을 막으면서 데이타 수신을 완료하는 시간을 앞당기기 위해서는 적절한 전달자를 선택하는 것이 중요하다. 이러한 선택의 정확도를 높이기 위해서는 타 노드들이 어떤 데이타를 갖고 있는지 정보를 확보하는 과정이 필요하다.

Informed delivery에서는 각 참여자는 지속적으로, 어떠한 데이타를 가지고 있는가 정리한 summary 정보를 교환한다. 이 과정에서 컨트롤 오버헤드가 발생하게 된다. 또한, 정보 교환 시 패킷 크기를 줄이기 위하여 모든 데이타 정보를 다 표현하는 대신 요약본 즉, summary를 사용하게 되는데 summary는 제한된 길이의 데이타로서 정확한 상태정보를 표현할 수 없기 때문에 필연적으로 데이타의 중복 수신이 발행하게 된다.

SplitStream에서는 데이타 전달 부담을 분산시키기 위해 단순한 트리가 아닌 forest를 제안하였다. 데이타 전달이 여러 개의 트리를 통해서 이루어지므로 개별 노드의 입장에서는 데이타 전달자가 복수 개 존재하게 된다. 송신자는 전송해야 할 데이타를 여러 stripe로 나누고, 각 stripe을 다른 트리를 통하여 전송한다. 각 노드는 서로 다른 트리의 부모 노드로부터 다른 stripe에 속한 데이타를 수신하므로 중복된 데이타를 수신하는 문제는 피할 수 있다. 그러나 복수 개의 트리를 관리해야 하는 부담이 남게 된다.

이중 멀티캐스트 메쉬 기법에서도 단일 전송 트리를 구성해서 데이타를 전송하는 것이 아니라 두 개의 멀티캐스트 그래프를 구성하여 데이타를 전송하는 경로를 추가로 확보하는 기법을 제안하고 있다. 그리고, 트리상의 노드들은 단순히 데이타만을 전달하는 것이 아니라 데이타를 디코딩하고 인코딩해서 새로운 데이타를 만들어 전달한다. 이 방법 역시 다중 전달 경로를 가지므로 데이타 전달 속도를 향상시킬 수 있지만, 매 노드에서 데이타를 디코딩해야 하는 프로세싱 부담을 갖는다.

P2P 기법에서 협력적인 파일 송수신 방법으로 BitTorrent[9]를 대표적인 예로 들 수 있다. 대규모 데이타를 250KB 단위의 조각으로 나누어서 개별 수신자가 여러 피어들로부터 서로 다른 조각을 수신하여 원래 데이타를 복원할 수 있도록 한다. 각 수신자는 자신의 수신 상태 정보를 traker에게 등록하고, tracker를 통해 다른 피어들이 어떤 조각을 갖고 있는지 정보를 획득하게 된다. 개별 피어들이 수신 정보를 교환하는 것이 아니라 중앙의 tracker에서 집중하므로 제어 메시지 부담은 감소하지만, tracker가 bottleneck이거나 오류의 중심이 될 수 있다는 단점을 가지며, 조각의 단위가 크다는 단점을 갖는다.

3. 제안하는 기법

3.1 개요

본 논문에서는 저장된 파일을 다운로드하는 상황이 아니라, 소프트웨어 갱신 등과 같이 특정 서버에서 다수의 사용자에게 '파일'을 배포하는 상황에서 적용할 수 있는 기법을 제안하고자 한다. 본 논문에서 제안하는 기법은 기존의 트리 기반 응용계층 멀티캐스트 기법을 확장하여, 패킷의 중복 수신을 피하면서 부모 노드 외의

추가적인 데이타 전달자를 선택하는 방법을 제안한다. 이 때, 각 노드의 입장에서, 부모 노드 외의 데이타 전달자를 가상 소스(virtual source)라고 칭하고, 해당 노드를 가상 소스로 하는 데이타 수신자를 가상 싱크(virtual sink)라고 칭한다. 가상 소스(virtual source)는 가상 소스 후보(virtual source candidates) 중 선택되는데, 가상 소스 후보는 해당 노드의 처음 수신 패킷 순서 번호보다 작은 처음 수신 패킷 순서 번호를 갖는 수신자들 중의 일부이다.

세션에 참여한 수신자는 트리의 적절한 곳에 위치하게 된다. 한 수신자가 세션 가입 후 부모로부터 첫 패킷을 받게 되면 원 데이타 송신자에게 자신의 주소와 처음 받은 패킷의 순서 번호를 알린다. 이와 동시에 가상 소스 후보 목록을 요청하게 된다. 원 데이타 송신자는 순서 번호의 차례에 따라 정렬하여 리스트를 갱신하면서, 해당 수신자의 처음 수신 패킷 순서 번호보다 작은 처음 수신 패킷 순서 번호를 갖는 수신자를 선택하여 응답한다. 따라서 가상 소스를 찾는데 소요되는 시간은 매우 작다. 수신자는 가상 소스 후보 목록 중에서 적당한 대상을 선택하여 가상 소스의 역할을 해 줄 것을 요청한다. 요청에 응락하는 답을 수신하게 되면, 해당 가상 소스가 세션에 머물러 있는 동안 해당 패킷을 수신할 수 있다. 가상 싱크의 역할을 하는 수신자도 다른 수신자들에 대해 가상 소스의 역할을 수행해야 한다. 한 수신자가 사용할 수 있는 네트웍 자원은 제한되어 있으므로, 각 수신자가 가질 수 있는 가상 소스의 수와 가상 싱크의 수는 제한된다.

그림 1에서 간단한 예를 보이고 있다. 수신자 O와 P는 각각 전송 트리 상의 부모 노드 외에 자신보다 일찍

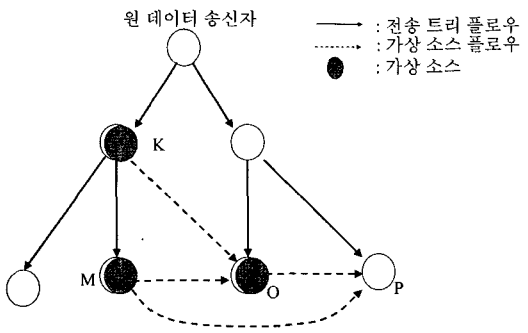


그림 1 제안하는 기법의 다중 전달송자 사례

세션에 가입한 수신자인 <K, M>과 <O>로부터 데이타를 수신함으로써 단위 시간 내에 수신하는 데이타의 개수를 늘릴 수 있다.

3.2 설계 고려 사항

본 논문에서 제안하는 기법을 설계하는 데 있어 고려한 중요한 가정은 크게 세 가지가 있다. 첫째, 사용자들은 특정 서버가 데이타를 응용 계층 멀티캐스트를 사용하여 배포한다는 것을 알고 있고, 데이타를 얻기 위해 해당 세션에 가입해고 서버의 데이타를 수신하는 단계로 데이타 전송이 이루어지는 상황에서 사용할 수 있는 기법을 제안한다. 해당 응용 계층 멀티캐스트 세션에 대한 정보를 획득하는 과정은 본 논문의 논의의 범위를 벗어나므로, 여기서 자세하게 언급하지 않는다.

둘째, 파일은 tornado codes [11]를 사용하여 인코딩되는 것으로 가정하여, 각 수신자는 데이타의 순서 번호가 일련되게 수신하는 것이 중요하지 않고, 서로 다른 패킷을 일정 개수 이상 수신하면 원래 파일을 복원할 수 있다. 즉, 단위 시간에 수신하는 데이타의 개수를 늘림으로써 성능을 향상시킬 수 있고, 데이타 수신을 완료하는데 소요되는 시간을 줄일 수 있다.

셋째, 그림 2에서 보이는 바와 같이, 수신자가 세션에 가입해서 처음으로 수신하는 패킷의 순서 번호는 서로 다르다는 것이다. 여기서, S_i는 수신자 i가 처음 수신한 데이타의 순서 번호를 의미한다. 이 경우, 수신자 i는 $[S_i, S_j)$ 사이의 데이타를 갖고 있고, 전송 트리만을 사용해서 전송한다고 할 때, 이 데이타들은 i 이후에 세션에 참여한 수신자들이 갖고 있지 않은 데이타이다. 마찬가지로, 수신자 m은 $[S_m, S_n)$ 사이의 데이타를 갖고 있으며, 이는 수신자 n, o, p, q는 갖고 있지 못한 데이타이다. 따라서 수신자 m이 $[S_m, S_n)$ 사이의 순서 번호를 갖는 데이타들을 n, o, p, q에 송신한다면, 중복되지 않는 데이타로 해당 수신자들의 수신량을 늘려주는 효과를 얻을 수 있다. 이 때, $[S_m, S_n)$ 사이의 순서 번호를 갖는 데이타들을 수신자 m의 *gift*라고 칭한다.

이러한 아이디어를 적용하기 위해서는 각 수신자가 다른 수신자들의 처음 수신 패킷 순서 번호를 알 수 있는 방법이 필요하다. 본 논문에서는 모든 수신자가 원 데이타 송신자의 주소 정보를 알고 있고, 원 데이타 송신자가 자신의 데이타를 누가 수신하고 활용하는가를 파악하기 원하는 서비스를 가정한다. 그러므로, 각 수신자가 원 데이타 송신자에게 자신의 처음 수신 패킷 순

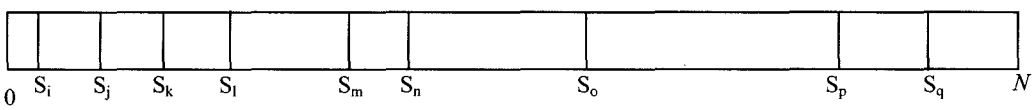


그림 2 전체 데이타 수신 번호 공간과 각 수신자의 처음 수신 패킷 순서 번호

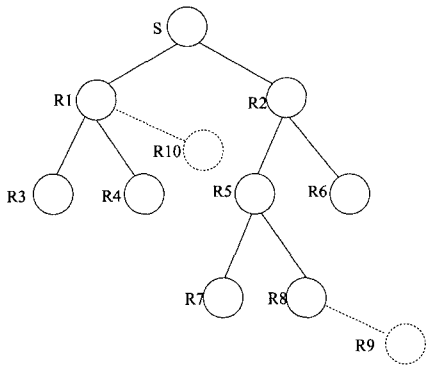


그림 3 가입 순서와 처음 수신 패킷 순서 번호 사이의 관계 사례 1

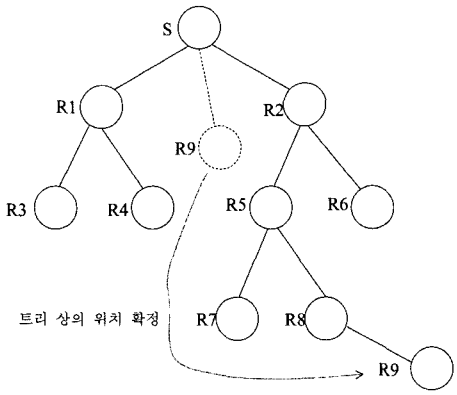


그림 4 가입 순서와 처음 수신 패킷 순서 번호 사이의 관계 사례 2

서 번호를 등록하고 이 정보를 원 데이타 송신자가 관장한다고 해도 무리가 없다.

수신자들의 처음 수신 패킷 순서 번호 정보를 관리할 때, 단순히 등록 메시지가 도착하는 순서에 따라 정보를 정렬하는 것은 바람직하지 않다. 수신자가 전송 트리 상의 어느 위치에 놓이게 되느냐에 따라 가입 순서와 첫 데이타의 순서 번호가 일치하지 않을 수 있다는 점이다. 그림 3의 예에서와 같이, 전송 트리가 구성되어 있는 상황에서, R9과 R10이 신규로 가입한다고 하면, R10이 R9보다 먼저 가입했다고 하더라도, 트리 상에서의 S와의 거리가 멀게 되므로, 가입 시 처음 수신하게 되는 패킷의 순서 번호가 R10보다 작을 수 있으며, R9에서 보낸 처음 수신 패킷 순서 번호가 R10이 보낸 처음 수신 패킷 순서 번호보다 늦게 도착할 수 있다. 그러므로, 처음 수신 패킷 순서 번호의 순서에 따라 수신 상태 정보가 관리되어야 한다.

원칙적으로 본 논문에서 제안하는 방식은 전송 트리를 구성하는 방법론과 무관하게 동작하는 것이 가능하지만, 각 수신자가 수신 상태를 등록하는 시기에 관련해서 특수하게 고려해야 하는 상황이 있다. 전송 트리를 생성하는 기법 중에는 세션에 가입하는 순간에는 원 데이타 송신자 S로부터 직접 데이타를 수신하다가 트리 상의 적절한 위치를 찾아서 부모 노드를 새롭게 설정하는 방식이 있다. 그림 3에서 보이는 경우와 같이, 전송 트리의 끝단에 위치하게 되어 새 부모 노드로부터 수신하는 데이타의 순서 번호가 S로부터 수신하던 데이타의 순서 번호에 비해 늦을 수도 있다. 이러한 경우에는, 트리 상의 위치를 확정한 후에 수신 상태 정보를 송신자에게 제공하는 것으로 한다.

3.3 상세 알고리즘

3.3.1 파일 인코딩 및 전송

원 데이타 송신자는 파일을 FEC를 사용하여 인코딩하여 패킷을 생성해낸다고 가정하며 인코딩된 패킷은 기본적으로 트리를 따라 전달된다. 하나의 파일을 구성하는 패킷의 양을 Q라 하자. Q 개의 패킷을 인코딩함으로서 N개의 서로 다른 패킷이 만들어진다. $N=Q+L$로 표현될 수 있는데, L은 잉여 패킷의 개수이다. 디코딩 오버헤드는 ε로 나타낼 수 있으며, N개의 인코딩된 패킷 중 $(1+\varepsilon)Q$ 개의 서로 다른 패킷을 받게 되면 파일을 복구해 낼 수 있다. $N=CQ$ 로 표현될 수도 있는데 여기에서 C는 stretch factor를 의미한다. C가 커짐에 따라 패킷의 중복 수신 가능성은 떨어지지만 인코딩/디코딩 오버헤드는 증가한다. 마지막 $(N-1)$번째 패킷을 전송한 후에 원 데이타 송신자는 0번 패킷부터 다시 전송하게 되는데 이를 싸이클이라 한다.

가상 소스는 자신의 *gift* 구간 내의 패킷만을 가상 싱크에게 전송하게 되는데, 한 싸이클 내에서 가상 소스간 *gift*는 서로소이기 때문에, FEC 코딩에 적절한 stretch factor가 사용되었을 경우 패킷의 중복 수신은 거의 일어나지 않게 된다.

3.3.2 처음 수신 패킷 순서 번호 정보 관리

원 데이타 송신자는 세션의 시작과 끝을 관장하며 수신자 리스트를 유지하고 관리한다. 리스트의 각 엔트리는 <*ma*, *fs*, *ls*>과 같은 튜플로 구성된다. *ma* 는 수신자의 주소, *fs*는 해당 수신자가 등록한 처음 수신 패킷 순서 번호이며, *ls*는 다른 수신자의 *fs*로서 해당 수신자의 *fs*보다 크되 가장 작은 값이 된다. 해당 수신자의 *gift* 내의 패킷 순서 번호는 [*fs*, *ls*) 안에 있다.

리스트는 그림 5에서 볼 수 있듯이 *fs*에 의하여 정렬된다. *ls*는 리스트에 해당 수신자를 위한 entry가 만들어질 때는 결정되지 않고, 새로운 entry가 추가될 때 그 값이 결정되고, 추가로 수정이 가능하다. 각 수신자는 자신의 *gift*의 범위를 독자적으로 파악할 수 없고, 원 데

Member address (*ma*)	*fs*	*ls*
220.0.11.92	0	10
220.34.6.122	10	21
210.184.23.5	21	43
…	…	…
210.107.192.210	345	

그림 5 원 데이타 송신자에 의해 유지되는 수신자 리스트의 예

이타 송신자로부터 통지를 받아야 한다. 해당 수신자의 *ls*가 새로 결정되거나 변경되는 경우 원 데이타 송신자는 해당 수신자에게 통지한다.

수신자 *k*가 세션에 가입하여 전송 트리에 위치를 확정해서 데이타를 수신하게 되면, *msg_first_packet*에 처음 수신 패킷 순서 번호를 실어서 원 데이타 송신자에게 전송한다. 이 메시지를 수신하면, 원 데이타 송신자는 해당 수신자를 위한 entry를 생성하고, 아직 *ls*가 결정되지 않은 수신자가 있거나 지금 알려진 처음 수신 패킷 순서 번호에 의해 *ls*가 수정되어야 하는 수신자가 있는지 점검한다. 리스트의 마지막 entry부터 거꾸로 점검하면서 갱신하면 되므로 갱신에 소요되는 시간은 길지 않다. *ls*가 갱신된 수신자들에게 *gift*가 결정 혹은 변경되었다는 것을 통지하기 위해 *msg_gift*를 해당 수신자들에게 전송한다.

수신자는 자신이 원 파일을 복구할 만큼의 충분한 데이타를 수신하고 나면 세션을 탈퇴한다. 이 때, 세션에서 탈퇴하기 전에 원 데이타 송신자에게 *msg_exit*를 보내게 되며, 원 데이타 송신자의 수신자 리스트에서 탈퇴하는 수신자에 대한 정보는 삭제된다. 이 때, 알고리즘의 복잡도를 높이지 않기 위해, *ls*의 추가적인 수정은 발생시키지 않는다.

3.3.3 가상 소스와 가상 싱크의 상호작용

각 수신자는 *msg_first_packet*를 원 데이타 송신자에게 전송하고 나서, *msg_candidates_request*를 다시 원 데이타 송신자에게 전송하여 가상 소스 후보 목록을 요청한다. 이 *msg_candidates_request*에 원하는 가상 소스 후보의 최대 개수 δ를 명시한다. δ의 구체적인 값은 각 수신자의 네트워크 환경에 따라 달라진다. 각 수신자는 사용 가능한 네트워크 대역폭의 정확한 값을 알지 못한다 하더라도, 네트워크 환경 설정 정보 등을 통하여 추정치를 가질 수 있으므로 이에 근거하여 δ의 값을 결정할 수 있다고 가정한다.

원 데이타 송신자는 *fs* 순서대로 정렬되어 있는 가상 소스 후보 목록을 *msg_candidates_reply*를 통해서 전달한다. 가상 소스 후보 목록을 받은 수신자는 목록상의 후보들 중에서 가상 소스 역할을 요청할 대상을 선택하는데, 이 때 *fs*의 역순으로 대상을 선택한다. 선택된 대상에게 *msg_vsource_join_request*를 전송하여 *gift* 전송을 요청한다. *msg_vsource_join_request*를 수신한 수신자들은 *msg_vsource_join_reply*를 요청자에게 전송하여 *gift*의 전송의 가부를 알리게 된다. 이 때, 가상 소스 후보는 자신의 out_degree를 점검하여 자신의 네트워크 대역폭 범위 안에서 out_degree를 증가시키는 것에 무리가 없는지 파악하고 *gift* 전송 여부를 결정한다. 이미 δ의 값을 결정하는 경우와 마찬가지로, 각 수신자는 사용 가능한 네트워크 대역폭의 정확한 값을 알지 못한다 하더라도 추정치를 가질 수 있으므로 이에 근거하여 out_degree의 최대값 γ를 결정한다.

전송을 허락하는 경우에는, 요청자가 가상 싱크가 되고, 전송을 허락한 수신자가 가상 소스가 된다. 가상 싱크가 가상 소스에게 *gift* 수신을 위한 TCP 연결을 설정할 것이고, 이 TCP 연결을 사용하여 가상 소스가 *gift*에 포함되는 패킷들을 차례대로 전송한다. 가상 소스는 자신이 원 파일을 재생할 수 있는 만큼의 데이타를 모두 수신하고 난 후에도 자신의 가상 싱크들에게 *gift*에 있는 패킷들을 전송하기 위해 세션을 탈퇴하지 않고 세션에 머물러 있을 수 있다.

각 수신자는 가상 소스나 가상 싱크의 오류(failure)를 파악하기 위하여 정기적으로 probing을 하게 된다. 오류가 발생한 수신자의 엔트리는 원 데이타 송신자의 수신자 리스트에서 삭제된다. 또한 오류가 발생한 수신자의 가상 싱크들은 새로운 가상 소스를 선택해야 한다.

3.3.4 전진/후진(moving forward / moving backward) 알고리즘

수신자 *k*가 전송 트리 상의 부모 노드로부터 수신하는 패킷은 순서 번호가 *fs*에 비해 큰 값으로만 증가해 가므로 이를 '전진'이라고 칭한다. 그리고, 가상 소스를 자신의 *fs*보다 작은 *fs*를 갖는 수신자들에 대해 역순으로 선택하게 되므로 이를 '후진'이라고 칭한다. 그림 6에서 가상 소스를 모집하는 과정을 임시 코드의 형태로 보이고 있다.

Recruiting the virtual sources (*k*th member's view)

```
1) backward=0, in_degree=0
2) VSL=[] // virtual source list
3) while in_degree < δ and (k – backward) > 0:
4)    backward++
5)    if (k - backward)th member accepts the request for gift:
6)       VSL = VSL ∪ (k - backward)
7)       in_degree++
```

그림 6 가상 소스를 선택하는 과정

그림 7에서 보이는 바와 같이, 각 수신자는 자신보다 *fs*가 작은 수신자들로부터 *gift*에 해당하는 패킷들을 수신하므로, 즉, 후진과 전진을 동시에 진행함으로써 원 데이타 복원을 위해 필요한 개수만큼의 패킷을 확보하는 시간을 단축할 수 있다. 그리고, 자신보다 *fs*가 큰 수신자들에게 자신의 *gift*를 전송함으로써 타 수신자가 원래 데이타 복원에 필요한 패킷을 확보하는 시간을 단축시켜 준다.

가상 소스가 자신의 *gift*를 가상 싱크에게 모두 전송하고 나면 가상 싱크에게 *msg_exhaust*를 전송한다. 이 메시지를 받은 가상 싱크는 in_degree를 1만큼 감소시키고 새로운 가상 소스를 선택하여 *msg_vsource_join_request*를 전송한다. 이 때는 원 데이타 송신자로부터 받은 가상 소스 후보 목록에 있는 수신자로서 아직 가상 소스로서 활용되지 않았으면서 *fs*가 가장 큰 수신자로 선택한다. 만약 현재 가지고 있는 가상 소스 후보 목록에 있는 모든 가상 소스를 다 활용했다면, 원 데이타 송신자에게 *msg_candidates_request* 메시지를 보내는 과정을 반복한다. 이 때, 동일한 후보 목록을 받는 상황을 피하기 위해 몇 번째 요청인지를 표시한다.

파일 복구에 충분한 개수의 패킷을 받은 후에는 모든 가상 소스에게 *msg_remove*를 보내게 된다. 이 메시지를 수신한 가상 소스는 TCP 연결을 끊고 out_degree를 1만큼 감소시킨다.

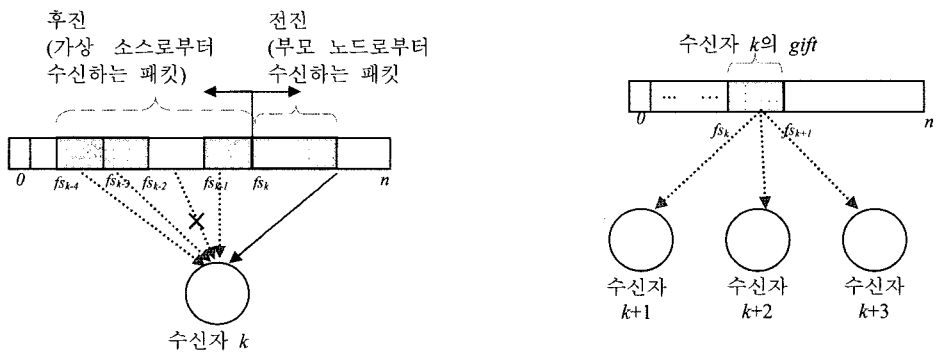
원 데이타 복원을 끝내고 세션에서 탈퇴할 때에는, 모든 가상 싱크에게 *msg_remove*를 전송한다. 이 때는 이미 가상 소스와의 연결은 끊어진 상태이므로 가상 싱크에게만 통지하면 된다.

4. 성능 평가

본 장에서는, 편의를 위해, 제안하는 기법을 OMFVS (Overlay Multicast for File delivery with Virtual Sources)라 칭한다. OMFVS는 응용 계층 멀티캐스트 기법의 시험 개발 환경인 MACEDON [12]을 이용하여 설계하고 구현되었다. MACEDON을 사용하면, 상태 전이도만 설계하여 입력하면 프로토콜을 구현할 수 있어 구현에 드는 노력을 크게 절감할 수 있다. 구현물을 전 세계적인 오버레이 실험망인 PlanetLab에 설치하여 성능을 평가하였다. 기존의 다중 전달송자 응용 계층 멀티캐스트 기법인 Bullet과 성능 비교를 하였으며 데이타 전송을 위해서 TCP를 사용하였고, 제어 메시지를 교환하는 데는 연결 설정 부담 등을 줄이기 위해 MACEDON이 제공하는 reliable UDP 기능을 사용하였다. 실험에 적용된 파라미터는 표 1과 같다.

4.1 컨트롤 오버헤드

그림 8과 9는 각 수신자가 발생하는 제어 메시지의 평균 개수 및 byte로 표현된 양을 각각 보이고 있다. 파일 수신을 마친 후에도 세션에 남아 다른 수신자들에게 데이타를 전송해주는 수신자들이 있다. 이렇게 파일 수신을 완료하고 나서 세션을 빠져나가기 전까지 다른 수



(a) Virtual sink (b) Virtual source

그림 7 가상 싱크와 가상 소스의 gift 수신 및 전송

표 1 실험 환경 변수의 값과 의미

변수	값	상세 설명
수신자의 개수	50	아시아 5, 유럽 4, 북미 39, 남미 1, 호주 1
δ	10	In-degree의 최대값
γ	10	out-degree의 최대값
원 데이타 크기	20 MB	
C	10	FEC의 stretching factor
ε	0.05	FEC decoding overhead

신자들에게 데이타를 전송하는 시간을 '기여 지연 시간(contributing time)'이라 한다.

발생되는 제어 메시지의 개수 관점에서 보면 제안하는 기법은 Bullet이 발생시키는 제어 메시지의 10%만을 발생시킨다. 제어 메시지의 양의 측면에서는 0.16% 만큼만 발생시키고 있다. Bullet의 경우에는 모든 수신자가 자신의 수신 상태를 주기적으로 공지하는 용도로 제어 메시지를 발생시키므로 본 논문에서 제안하는 기법에 비해 전송하는 제어 메시지의 수와 양이 크다. 그러나, 제안하는 기법에서는 자신의 처음 수신 패킷 순서 번호를 등록하는 과정과 가상 소스 목록을 요청하고 수신하는 과정, 그리고 가상 소스들에게 요청하는 과정, 가상 싱크의 요청에 응답하는 과정 등에서 제어 메시지를 사용한다. 즉, 각 수신자마다 50개 이내의 제어 메시지를 발생시키므로, 세션의 길이가 길어질수록 전체적으로 제어 메시지의 개수나 양의 측면에서 Bullet보다 훨씬 적게 된다.

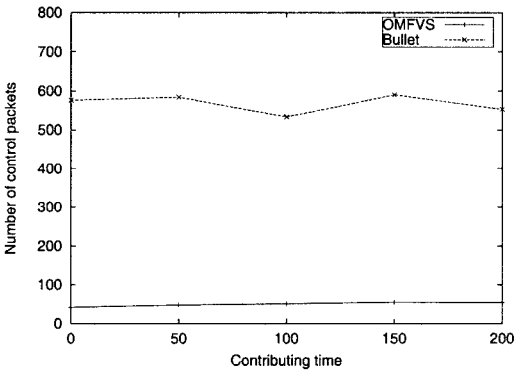


그림 8 수신자당 평균 제어 메시지 발생 개수

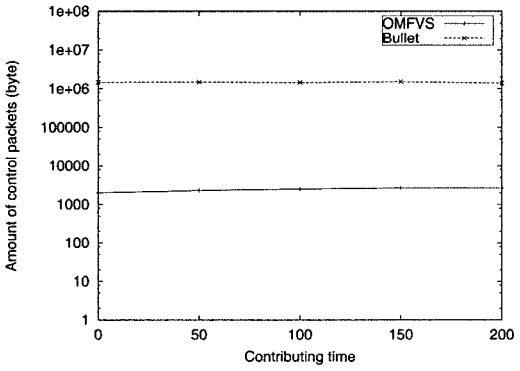


그림 9 수신자당 평균 제어 메시지 발생량 (byte 수)

4.2 패킷 중복율

각 수신자가 전체 수신한 데이타 패킷의 개수와 순서 번호가 중복되는 데이타 패킷의 개수의 비율을 관찰한 결과, Bullet은 4% 정도에 해당하는 패킷들이 중복 수신된다. 10MB의 데이타를 수신한다고 할 때, 400KB 정도의 데이타가 중복 수신된다는 것이다. 하지만 제안한 기법에서는 패킷의 중복이 전혀 발생하지 않았다.

4.3 완료 지연 시간

세션에 가입한 때부터 파일 수신을 완료하는 시점까지 걸린 시간을 '완료 지연 시간(completion time)'이라 한다. 실험을 위하여 임의의 멤버는 파일 수신을 완료하는 즉시 세션에서 탈퇴함을 가정한다. 그림 10은 세션이 진행되어 감에 따라, 즉 세션에 가입하는 수신자의 수가 증가함에 따라 평균 완료 지연 시간의 누적 평균이 어떻게 변해 가는지 보이고 있다. 세션의 초기에는 Bullet의 제안하는 기법에 비해 평균 완료 지연 시간이 작으나 시간이 지남에 따라 제안한 기법의 평균 완료 지연 시간이 더 낮아짐을 볼 수 있다.

그림 11은 세션 크기 즉 참가자의 수에 따른 완료 지연 시간의 변화를 보이고 있다. Bullet과 제안하는 기법과의 차이를 보다 효율적으로 보이기 위해, 제안하는 기법의 완료 지연 시간과 Bullet의 완료 지연 시간 사이의

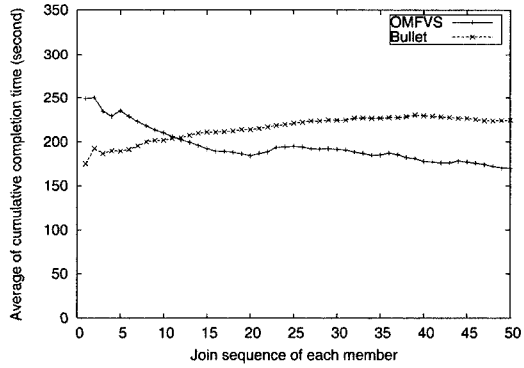


그림 10 완료 지연 시간의 누적 평균

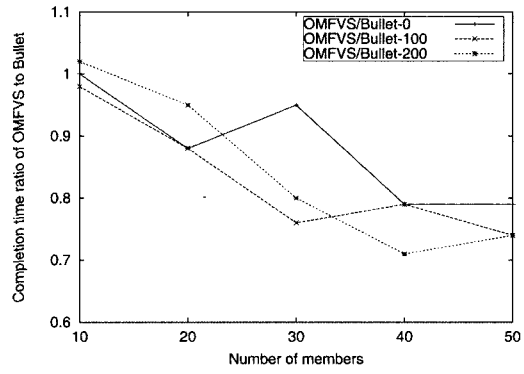


그림 11 Bullet 대비 OMFVS의 완료 지연 시간 비율

비율로 표현하였다. 세션 크기가 커질수록 그 비율이 점점 더 작아지는 것을 볼 수 있다. 즉, 제안하는 기법의 완료 지연 시간이 Bullet의 완료 지연 시간에 비해 크게 작아지고 있다는 것을 의미한다. 예를 들어, 세션 크기가 50이었을 경우 제안한 기법은 Bullet에 비해 20 - 30% 가량의 완료 지연 시간 절감 효과를 보이는 것을 알 수 있다.

이러한 현상들이 보이는 이유는, 본 논문에서 제안하는 기법이 처음 수신 패킷 순서 번호 간의 차이를 가상 소스 선택 조건으로 사용함으로써, 초기에 가입하는 수신자보다는 후기에 가입하는 수신자가 가상 소스를 택할 수 있는 범위가 많아지므로 본 논문에서 제안하는 기법의 장점을 더 잘 활용할 수 있기 때문이다. 그리고, 세션의 크기가 커지면, 그만큼 가상 소스를 선택할 수 있는 범위도 커지게 되므로 더 다양한 수신자들을 가상 소스로 활용하여 데이타 수신량을 효율적으로 늘릴 수 있기 때문이다.

5. 결 론

일대다 스트리밍 서비스, 혹은 일대다 파일 전송 서비스 등을 위해 IP 멀티캐스트의 대안으로 응용 계층 멀티캐스트가 고려되고 있다. 그런데, 트리 구조의 전송 방식을 채택함으로 인해서 성능이 낮은 수신자가 전송 트리 상의 자손 노드 전체의 성능을 낮추는 단점을 갖고 있다. 이를 해결하기 위한 방안으로 다중 트리를 생성하거나 전송 트리 상의 부모 노드 외에 추가적인 전달자를 채택하여 단위 시간 내에 수신할 수 있는 데이타의 양을 늘려서 전체 성능을 향상시키는 방안들이 많이 제시되고 있다. 그런데, 이런 방법들은 각 수신자의 수신 상태를 파악하기 위해 수신자들 간에 교환해야 하는 제어 메시지의 부담을 갖는다.

본 논문에서는 이러한 단점을 해결하기 위한 가상 소스 선택 기법을 제안하였다. 본 논문에서 제안하는 기법은 수신자들이 가입하는 시점 사이에 차이가 있다는 현상에 기반하여 처음 수신 패킷 순서 번호의 순서에 따라 가상 소스를 선택하게 함으로써 수신 상태 정보를 교환하는 부담을 배제하였다. 기존에 제안된 Bullet과의 성능 비교에서, 세션이 진행되어 감에 따라 세션 참가자의 수가 많아지면서 본 논문에서 제안하는 기법이 우수한 성능을 보임을 확인할 수 있었으며, 중복해서 수신되는 패킷의 수가 월등히 적게 나타나는 것을 확인할 수 있었다.

본 논문에서 제안하는 기법은 특정한 트리 구성 기법을 가정하는 것이 아니므로, 기존의 트리 기반 응용 계층 멀티캐스트 기법에 추가적으로 적용하여 활용하는 것이 가능하다. 추후 다양한 트리 기반 기법에 적용하여 그 실효성을 확인해 볼 것이며, 다양한 세션 가입 패턴을 가정한 환경에서의 실험을 통하여 실용성을 높이기 위한 연구를 지속할 것이다.

참 고 문 헌

[1] L. Sahasrabuddhe, B. Mukherjee, "Multicast routing algorithms and protocols: a tutorial," IEEE Network, Volume 14, Issue 1, Jan.-Feb. 2000, pp. 90-102.

[2] Christophe Diot, Brian Neil Levine, Bryan Lyles, Hassan Kassem, Doug Balensiefen, "Deployment Issues for the IP Multicast Service and Architecture," IEEE Network, Volume 14, Issue 1, Jan.-Feb. 2000, pp. 78-88.

[3] A. Ganjam, H.Zhang, "Internet multicast video delivery," Proceedings of the IEEE, Volume 93, Issue 1, Jan 2005, pp. 159-170.

[4] Y.-H.Chu, S.G.Rao, S.Seshan, and H. Zhang, "A case for end system multicast," IEEE JSAC, Sp. Issue on Network, Vol. 20, No.8, October 2002, pp. 1456-1471.

[5] D. Kostic, A. Rodriguez, J. Albrecht, and A. Vahdat, "Bullet: High Bandwidth Data Dissemination Using an Overlay Mesh," In Proc. of ACM Symposium on Operating Systems Principles (SOSP), October 2003.

[6] J. Byers, J. Considine, M. Mitzenmacher, and S. Rost, "Informed Content Delivery Across Adaptive Overlay Networks," In Proc. of ACM SIGCOMM, August 2002.

[7] M. Castro, P. Druschel, A. Kermarrec, A. Nandi, A. Rowstron, and A. Singh, "SplitStream: High-bandwidth content distribution in a cooperative environment," In Proc. of International Workshop on Peer-to-Peer Systems (IPTPS), February 2003.

[8] Y. Zhu, B. Li, and J. Guo, "Multicast with Network Coding in Application-Layer Overlay Networks," IEEE Journal of Selected Areas in Communications (JSAC), Vol.22, No.1, January 2004, pp. 107-120.

[9] B. Cohen, "Incentives build robustness in bittorrent," In Proc. of Workshop on Economics of Peer-to-Peer Systems, May 2003.

[10] Scott Karlin, "PlanetLab: A Blueprint for Introducing Disruptive Technology into the Internet," joint Princeton ACM / IEEE Computer Society meeting, November 2003.

[11] J. Byers, M. Luby, M. Mitzenmacher, and A. Rege, "A digital fountain approach to reliable distribution of bulk data," In Proc. of ACM SIGCOMM, 1998.

[12] A. Rodriguez, S. Bhat, C. Killian, D.Kostić, and A.Vahdat, "MACEDON: Methodology for automatically Creating, Evaluating, and Designing Over-

lay Networks," Technical Report CS-2003-09, Duke University, July 2003.

이 수 전

1995년 3월~2003년 2월 고려대학교 컴퓨터학과 학사. 2003년 3월~2005년 2월 한국정보통신대학교 공학부 석사. 현재 한국전자통신연구원 디지털방송연구단 위성관제팀 연구원. 관심분야는 Mission planning for satellite operations (위성운용 임무계획)

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

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