다자간의 통신환경에서 다양한 수신품질을 고려한 One-Time 오버레이 멀티캐스트 기법에 관한 연구

정회원 윤 미 연*, 김 기 영**, 김 대 원*, 신 용 태***

One-Time Overlay Multicast Techniques Considering Receipt Quality for m-to-n Comunication over Large Internet

Mi-youn Yoon*, Ki-Young Kim**, Dae-Won Kim*, Yong-Tae Shin*** *Regular Members*

요 약

오버레이 멀티캐스트 기법은 기존의 IP 멀티캐스트에서의 하드웨어적 문제를 해결할 수 있는 기법으로 최근 들어 활발한 연구가 진행 중에 있다 그러나 호스트간의 가상의 네트워크를 구성하여 그 위에서 전송 트리를 구성하기 때문에 멀티미디어 데이터의 실시간 전송의 경우 해당 멤버 중 가장 낮은 품질의 수신환경을 가진 호스트에게 동기화가 된다는 문제점을 가지고 있다 따라서 본 논문에서는 다자간의 통신환경에서 다양한 수신 환경을 지원하는 One-Time 오버레이 멀티캐스트 트리 구성 기법에 대해 제안한다 본 논문은 각 수신자가 자신의 만족할 만한 수신 품질로 데이터를 전송받음을 증명하고 트리는 구성을 위한 제어정보의 오버헤드가 적고 구성 시간 복잡도가 짧음을 보임으로써 다자간의 환경에서 동적으로 트리가 생성 소멸 가능함을 보였다 마지막으로 대규모의 인터넷 상에서 운영 가능하므로 보이기 위해 본 트리 구성 기법의 확장성이 우수함을 보였다

Key Words End System Multicast, Overlay Multicast, QoS, Reliable Multicast

ABSTRACT

IP Multicast has not been deployed because of hardware problems So a new scheme that is called Overlay Multicast for group communication has been emerged It supports IP Multicast functions, which is located on application level For developing it, we have been focused on efficient overlay tree construction among group members with low stretch and stress However, we should consider a variety of transmission or receipt condition since a real internet environment has users with various transmission/receipt rates. Thus, we make one-time source specific tree depending on required bandwidth informationof group members when a member requests data transmission Our mechanism provides satisfied data quality limited maximum transmission rate of the source to each group members Furthermore, we manage a large group enough as distributing control information to cores that are designated membersfor maintaining host member information Lastly, we prove that our tree guarantees data quality to each group members, and show low tree construction time is required In addition, for evaluating group scalability, we analyze control information increasing rate via group size, and validate its scalability

Ⅰ. 서 론

네트워크의 자원을 효율적으로 사용하기 위한 대표적인 라우팅 기술인 IP Multicast[4]는 인터넷 계층 자원의 효율적 사용이라는 최대의 장점에도 불구하고 실제 운용이 불가하다는 문제로 현재의 인터넷에 활

* 숭실대학교 컴퓨터학과 컴퓨터통신연구실({myyoon, kdwon2002}@cherry ssu ac kr),
**서일대학 정보통신계열 소프트웨어 전임강사(ganet89@seoil ac kr),
*** 숭실대학교 정보과학대학 컴퓨터공학부(shin@computing ssu ac kr)
논문번호 KICS2004-07-077, 접수일자 2004년 7월 4일

용되지 못하고 있다 이를 위한 대안으로 현재의 인프라 위에 멀티캐스팅 기법을 활용하여 다 대 다 또는 일 대 다 환경에서의 통신을 효율적으로하고자 하는 노력이 진행 중에 있다 응용계층 멀티캐스트 기법이 이 중의 하나이다

응용계층 멀티캐스트 기법은 기존의 인터넷 계층에서의 라우팅 트리를 중간 라우터의 도움 없이 가상의 위상을 구성하여 그 위상 위에서 호스트간의 라우팅 트리를 구성하여 멀티캐스트팅이 가능하도록 하는 기법이다 본 기법은 오버레이 멀티캐스트 기법이라고도 불린다. 트리를 구성하는 기본 조건으로서 스트레치(Stretch)[8] 와 스트레스(Stress)[8] 두 가지 요건을 충족할 수 있는 전송 트리를 구성하는 것에 주안점을 두어 연구되고 있다 이를 위해 그룹관리를 위한 제어위상과 전송위상 두 가지로 나누어 다자간의 통신을 지원하는 메커니즘이 제안되었다. 현재 제안된 응용계층 멀티캐스트 기법은 라우팅을 위한 메트릭을 종단간 지연시간이나 최단경로를 사용하고 있으며 이를 위한 그룹관리 기법을 제안 하였다

그러나 현재의 인터넷 환경이 대용량 멀티미디어 콘텐츠의 전송을 요구하고 실시간 스트리밍 서비스가 증가되고 있는 상황임에도 불구하고 이러한 기법은 대부분 기존의 라우팅 메트릭을 종단간 지연시간이나 최단경로를 사용하고 있기때문에 사용자의 요구 품질을 만족할 수 없음은 자명하다 [8]에서는 가상위상에서의 라우팅 메트릭을 대역폭 및 지연시간 정보를 이용함으로써 이러한 문제점을 해결하려고 했으나 강건한 메시를 구성하고 그룹에 가입한 멤버들이 해당 그룹의 모든 멤버의 정보를 갖고 있는 상태에서 기존의 멀티캐스트 라우팅 기법에 따라 일-대- 다 소스 기반의 멀티캐스트 트리를 구성하기 때문에 대규모의 인터넷 환경에서는 적용이 어렵다 또한 데이터를 전송하기 위해서는 전송 트리가 미리 구성되어야 하는 오버헤드를 가지고 있으며 새로운 멤버의 가입 또는 기존 멤버의 탈퇴 등에 따른 트리의 변경을 위해서는 제어메시지 교환비용이 그룹의 가입자 수에 비례하여 증가한다. [8]의 저자도 소규모의 네트워크에서 효율적으로 동작함으로 언급하고 있다.

따라서 본 논문에서는 사용자에게 만족할 만한 요구품질을 제공하고 대규모 인터넷환경상에서 적용 가능하며, 전송 트리의 구성 방식이 여러 제안된 방식과 비교하였을 때 전송 트리 형성시의 제어 메시지 비용을 줄이고, 시간복잡도 또한 높지 않은 비교적 단순한 멀티캐스트 기법을 제안하고자 한다. 즉 인터넷 상에서 적용될 수 있도록 확장성 있는 그룹 관리와 함께 다-대-다 환경에서 사용자 요구품질에 따라 동적으로 전송되는 전송 트리 기법을 제안하고자 한다

2장에서는 현재 제안된 응용계층 멀티캐스트 기법을 트리 우선 방식, 메시 우선 방식, 기타 방식으로 그리고 본 논문에서 초점을 맞추고 있는 대역폭을 고려하나 멀티캐스팅 기법으로 분류하여 소개하고 각 기법의 한계점을 제시한다 3장에서는 다자간의 환경에서 실시간 멀티미디어 통신에 적합한 One-Time 오버레이 트리 구성 기법을 제안하고 이를 위한 확장성 있는 그룹관리 기법을 제안한다. 4장에서는 제안하는 알고리즘에 따라 생성되는 제어 위상에서는 항상 다양한 수신 환경에서 수신자가 수신할 수 있는 품질을 보장함을 증명하고 트리의 생성 시간이 길지 않음을 보인다. 또한 그룹관리의 확장성을 보이기 위해 그룹의 멤버가 증가함에 따라 발생하는 제어 메시지의 증가도를 보이도록 한다. 마지막으로 5장에서는 결론과 함께 향후 연구방향을 제안한다

Ⅱ. 관련연구

오버레이 멀티캐스트기법에서는 각 호스트가 그룹관리 및 라우팅 기능 모두 수행한다 이를 위해서 각 호스트는 호스트간의 가상 네트워크를 구성하는 'Self-Organızaion'기능과 전송을 위한 네트워크 설정 기능- 'Self-Configuration' -을 갖추고 있어야 한다 이를 위해서 제어위상과 전송위상을 가지고 있다 이는 실제 IP계층에서의 라우팅 기능을 오버레이에서 수행하기 때문에 가상적인 위상이 필요하게 된다 제어위상은 호스트간의 그룹관리를 위해 필요한 위상이다. 그룹의 가입 및 탈퇴 또는 호스트 실패 등의 멀티캐스트 통신을 위한 기본 제어 메커니즘을 위해 형성된 위상이다 전송위상은 가상적으로 호스트간의 라우팅 경로가 구성된 것이다 따라서, 제어위상은 궁극적으로 전송위상을 위한 기본 위상이라고 볼 수 있다.

제안된 오버레이 멀티캐스트기법은 크게 트리우선 방식, 메시우선 방식 그리고 기타 방식으로 구성된다. 트리우선 방식은 제어위상과 전송위상 모두 호스트간의 구성된 트리를 통해서 수행되는 방식을 의미한다. 메시우선 방식은 제어위상으로서 풍부한 메시를 구성하고, 그 위에서 전송을 위한 트리를 생성하는 방식이다 기타 방식은 두 가지 형태를 벗어난 형태로서, 최단 거리 또는 지연시간이 아닌, 콘텐츠 정보를 이용한 멀티캐스트 방식[7,18] 또는 확장성 있는 그룹관리를 위한 멀티캐스트 방식[3] 등이 있다. 각각의 언급된

방법을 간략하게 알아보고, 다-대-다 환경에 적용하고자 할때의 제한점과 대용량 멀티미디어 콘텐츠 전송에 적용시의 한계점을 제시한다. 또한 대역폭 정보를 이용한 다양한 계층에서의 멀티캐스트 기법을 소개하고, 대규모 인터넷상으로의 적용시 문제점을 제시한다

2.1 트리우선 방식

대표적인 프로토콜로 YOID[2], HMTP[17] 등이 있다. 이중 HMTP 프로토콜의 경우, IP멀티캐스트가 가능한 영역의 연결을 자동으로 해주고, IP multicast가 지원하지 않는 곳에서는 호스트간의 오버레이 멀티캐스트 트리를 구성함으로 멀티캐스트 서비스를 제공해 주는 프로토콜이다. 각 그룹의 멤버는 멀티캐스트 에이전트로 동작한다 그룹관리 및 전송이 모두 생성된 트리를 기반으로 하여 구성되기 때문에, 다-대-다 환경에서 적용하고자 할 때에는 공유 기반 트리기법은 삼각라우팅으로 인한 데이터 전달이 지연된다는 단점을 가지며 다양한 수신환경을 고려하지 않고 있다 따라서, 각각의 수신 환경에 맞는 받아들일 만한 품질을 제공할 수 없다. 또한, 그룹관리의 측면에서도 멤버의 가입 및 탈퇴 등으로 인한 위상의 빈번한 변화로 인한 관리비용이 크다.

2.2 메시우선 방식

대표적인 프로토콜로 Narada[1,8]가 있다 이는 메시 기반의 제어위상과 트리형태의 전송위상 두가지를 기준으로 멀티캐스트 전송을 수행한다. 메시는 그룹관리를 위해서 수행되며 이렇게 형성된 메시 위에 트리를 구성하도록 되어 있다 네트워크 리소스 자원이 풍부한 메시를 구성하고자, 멤버의 가입 및 탈퇴, 그리고 실패 등의 요인으로 인한 메시의 재구성하고자 주기적인 제어 메시지를 멤버간에 교환하여 자원이 풍부한 메시 및 네트워크 분할 등의 문제를 극복하고 있다 그룹관리를 매우 강건하게 수행하고 있음을 알 수 있다. 그러나, 그룹의 모든 멤버가 자신을 제외한 다른 멤버들의 상태정보를 기억하고 있어야 하기 때문에, 소규모 그룹에서만이 가능하다.

또한, 멤버간의 메시지를 전송하고자 형성된 메시 위에서 트리를 구성하는데 이때, 형성되는 트리는 소스 기반의 트리이며, 일-대-다 멀티캐스트 통신 환경을 제공한다 형성되는 전송트리는 스트레스와 스트레치를 적당히 줄일 수 있는 형태를 취한다 기본적인 Narada는 트리상의 호스트간의 최단거리 정보를 이용하여 트리가 구성된다 그러나, 본 프로토콜을 멀티미디어 통신에 적용한 경우[8]에는 라우팅 메트릭을 대역폭 정보와 지연정보를 이용하여 형성된 메시위에서 라우팅 트리를 생성한다. 본 방식은 제어위상으로 메시를 구성하고, 해당 메시간의 주기적인 메시 교환을 통해 그룹의 상태를 관리하고, 좀 더 풍부한 네트워크 리소스 자원을 확보하기 위한 노력을 제공한다 따라서, 강건한 그룹관리는 가능하나, 이렇게 구성된 제어위상 위해서 전송 트리를 구성하기 때문에, 확장성이 결여된다. 하지만 트리를 생성시 라우팅 메트릭을 어떤 것을 사용하느냐에 따라 고품질을 제공할 수 있다. 본 방식은 다-대-다 수신환경에 적용하였을 경우, 송신자에 따라 소스기반의 트리를 구성하는데, 그 구성하는 시간 및 제어 메시지의 오버헤드가 매우 크다는 단점을 가지고 있다 그리고 한번 구성된 트리를 다시 변경하기가 어렵다는 단점이 있다.

2.3 기타 방식

확장성을 고려한 멀티캐스트 통신 방식으로 NICE[3]가 있다 이는 멀티캐스트 전송을 하고자 트리를 생성하는 부분에 초점을 둔 것이 아니라, 확장성 있는 그룹관리 부분을 중점적으로 다루고 있다 확장성 있는 그룹관리를 위하여 계층간의 클러스터를 형성하는 기법을 사용한다. 그룹의 모든 멤버들은 '계층 0'에서는 '계층 1'에서의 클러스터 들에 의해 분할되어 관리되며, 이는 다시 '계층 2'에서 관리되는 클러스터들에 의해 관리된다 이러한 방식으로 최상위 계층의 클러스터에는 하나의 멤버만이 존재할 때까지 형성되며, 이 구조는 논리적인 구조이다 각 클러스터의 대표자로서 각 멤버들로부터 중간에 위치한 호스트를 선출되고, 각 계층의 대표자는 자신의 산하 클러스트들을 관리한다. 이렇게 함으로써 그룹의 관리를 분산시키는 효과를 얻을 수 있다 클러스터의 대표자가 탈퇴하거나 실패한 경우, 주기적인 제어 메시지를 통해 대표자가 없음을 알고 다시 선출한다 그러나, 이 방법은 그룹관리를 위해 구성되어야 하는 제어 위상과 그 제어방식이 복잡한 단점이 있다 본 방식은 그룹관리의 확장성에 초점을 두었으나, 그룹이 확장됨에 따라 그룹관리를 위한 트리의 깊이가 증가하여 그 관리 절차가 복잡해진다

다른 방식으로 라우팅 메트릭을 콘텐츠로서 수행하는 것이다 이는 기존의 라우팅 개념에서 벗어나, 각 호스트가 가지고 있는 콘텐츠의 유사성을 비교함으로써 서로 데이터를 교환하는 방식을 갖는다. 대표적인 프로토콜로서 Pastry[18], Scribe[7] 등이 있다. 혼합형 방식은 콘텐츠의 유사성을 비교하여 이를 기반으로

표 1. 기존 연구의 비교

	HMTP[17]	NARADA[8]	NICE[3]	대역폭을 고려한 IP Multicast전송기법[13,14]
수신환경고려	전송지연	전송지연 대역폭	전송지연	계층적 압축기법[13] MDC 인코딩 &path diversity[14]
트리 유지 오버헤드	세션종료시까지	세션종료시까지	세션종료시까지	세션종료시까지
트리방식	소스기반 트리	소스기반트리	소스기반트리	소스기반트리
그룹관리	- 자신의 자식 정보만을 유지 관리 - 트리형태를 유지하기 위한 가입,탈퇴 절차 요함	- 메쉬구성 - 각 노드는 다른 멤버들의 상태 정보를 저장, 주기적 갱신을 요함	- 계층적 클러스터링 기법 - 자신의 하위 계층의 노드정보만을 유지관리	- IGMP사용 - 다른 멤버 존재 유무 및 상태를 알 필요 없음.

라우팅 경로를 생성한다. 이는 물리적 거리나 홉수를 기반으로 하는 라우팅이 아닌, 물리적 근접도를 무시한 채, 콘텐츠간의 유사성만으로 아이디를 부여함으로써 해당 아이디를 가지고 서로 메시지를 교환하는 방식이다. 이 기법은 유사한 콘텐츠를 다른 호스트로부터 분산하여 수신받을 수 있다는 장점이 있으나, 물리적인 근접도를 고려하지 않았으며, 해당 콘텐츠를 전송해주는 호스트가 많아질 경우, 대역폭의 낭비를 초래할 수 있다. 따라서, 네트워크의 비용적인 측면에서는 효율적이지 못하다.

2.4 대역폭을 고려한 멀티캐스트 전송방식

한 그룹안의 서로 다른 환경을 가진 멤버들의 사용자 요구 품질을 만족하기 위하여 IP 멀티캐스트 환경, 오버레이 멀태캐스트 환경등 멀티캐스트 전분야에 걸쳐 오랜 연구가 진행되어 왔다. IP Multicast 환경에서는 이를 위하여 특별한 인코딩 기법인 계층적 압축[13] 기법을 도입함으로써 해결 노력 또는 MDC[14] 인코딩 기법을 활용한 해결 노력이 대표적이다. 계층적 압축기법은 전송될 데이터를 계층적으로 압축하여 다수의 채널로 전송함으로써 수신 노드들에게 다양한 대역폭을 제공한다. 그리고 MDC기법은 원본 데이터를 두 개의 스트림으로 생성하여 2개의 경로로 전송하며, 2개의 스트림을 모두 전송받은 경우에는 고품질을 제공할 수 있으며, 그렇지 않다 하더라도 스트림을 제공받을 수 있도록 한 기법이다. 그러나, 이러한 기법은 오버레이 멀티캐스트 상에서는 모든 호스트가 해당 스트림의 인·디코딩이 가능해야 한다는 제약을 가지고 있다. 특히, 계층적 압축기법을 오버레이 네트워크 상에 활용한 연구[16]가 있으나 이 경우 하이퍼큐브 구조를 구축하기가 어렵다. 또한 MDC기법을 이용하는 경우는 완전히 상이한 2개의 경로로 데이터를 보내야 하는 어려움이 있다. 또한 다자간의 통신일 경우에는 2배의 스트림이 네트워크 상에서 전송되어야 하는 단점이 있다. 오버레이 멀티캐스트 상에서는 멤버간의 서로 상이한 대역폭을 고려한 기법으로는 대표적으로 NARADA[8]가 있다. 본 기법은 그룹 멤버간의 전송 트리를 구성시 기존의 멀티캐스트 라우팅을 적용하나, 라우팅 메트릭을 대역폭과 전송지연을 고려함으로써 이루어진다. 그러나, 이는 사전에 제어위상이 강건히 구축이 되고, 본 정보를 바탕으로 소스 기반의 트리를 구성함으로 소규모의 그룹에서 효율적이며, 대규모의 그룹 환경으로의 적용은 어렵다.

2.5 기존 연구의 분석

본 절에서는 앞절에서 소개한 기존 연구[3,8,13,14,17]들을 대규모의 그룹환경에 적용에 따른 각 장단점과 수신환경의 고려 정도를 분석하고자 한다. 이를 위해서 다음과 같은 요소들을 고려하였다.

- 수신환경의 고려 : 상이한 수신환경을 고려하기 위해 사용한 기법
- 트리 유지 오버헤드 : 전송 트리의 유지하여 하는 기간
- 그룹관리를 위한 노드당 유지비용 : 그룹관리를 위해 사용되는 각 노드에서 필요한 저장소의 크기 정도, 제안된 관리기법

표 1에서와 같이 기존에 제안된 오버레이 멀티캐스트 방식은 수신환경을 고려하기 위한 메트릭으로 전송지연 값을 대부분 사용하였으며, 사용자의 다양한 수신률은 고려하지 않았음을 알 수 있다. NARADA의 경우는 대역폭 정보를 함께 사용함을 볼 수 있다. 그러나, 그룹관리 측면에서 그룹에 참여한 모든 노드가 다른 멤버의 들의 상태정보를 모두 갖고 있기 때문에서 대규모 그룹환경에서는 적합하지 않음을 알

수 있다 대역폭을 고려한 IP Multicast 전송기법의 경우, 그룹관리가 가장 효율적으로 되고 있지만, IP 멀티캐스트가 실제 인터넷 상에 구현되기가 어렵다는 한계점을 가지고 있다. NICE의 경우, 계층적 클러스터링 기법을 사용하여 그룹의 확장성에 기여하였으나, 그룹의 멤버가 늘어나면 늘어날 수록 그룹관리를 위한 계층 깊이가 또한 함께 증가함을 알 수 있다. 또한 표1에서의 모든 연구가 전송 트리를 세션 종료시까지 유지 해야 하고 소스기반의 트리를 구성하기 때문에 이를 다자간의 통신 환경에 적용시 유지해야 하는 트리의 개수도 함께 선형적으로 증가함을 알 수 있다.

본 논문에서는 제어위상과는 독립적인 P2P기반 전송 트리의 구성하고 그에 따른 제어메시지 교환 오버헤드도 줄이고, 요구시에만 빠르게 동적으로 생성되면서 동시에 데이터를 전송하는 기법을 제안하며, 전송 트리를 구성하기 위한 정보수집 및 그룹관리를 행하기 위한 확장성있는 관리기법을 제안한다

Ⅲ. 제안하는 One-Time 오버레이 멀티캐스트 통신기법

본 논문에서는 다자간의 통신환경에서 실시간 멀티미디어 데이터의 그룹통신을 가능하게 하는 응용 멀티캐스트 프로토콜을 제안한다 제안하는 멀티캐스트 프로토콜은 제어 위상과 데이터 전송 위상 두가지의 위상을 갖고 운영된다. 먼저 3.2절에서 서술할 계층적 제어위상을 형성하여 트리 생성시 필요한 정보를 수집한다. 수집된 정보는 세션관리자 및 코어들이 관리하며, 이를 이용하여 각 호스트들은 제어위상과는 독립적인 전송위상을 형성한다 제어위상과 전송위상간의 연관성을 제거하고 계층적 그룹관리를 지원함으로써 확장성을 제공하였다 게다가, 데이터를 전송하고자 하는 호스트의 요구시 동적으로 One-Time 트리를 생성하게 함으로써 다자간의 환경에 적합하도록 하였다

3 1 One-Time 트리 생성 알고리즘

One-Time 트리는 제어 위상으로서 형성된 메시를 통해서 수집된 정보를 이용하여 실시간 스트림을 보내고자 원하는 노드가 자신이 소스가 되어 소스 기반의 트리를 구성함으로써 생성된다. 트리를 구성하는 알고리즘은 그림 1와 같다. One-Time트리를 구성하기 전에 그룹의 크기는 그룹의 루트가 될 소스가 3.2절에서 자세히 설명할 세션 관리자로부터 수식(1)를 통하여 정확한 그룹의 크기가 아닌 근사치인 그룹의 크기 |G|정보를 요청 및 수신하여 미리 알고 있으며 본 정보를 트리생성시 사용되는 'Data_REQ'메시지에 실려서 각 멤버에게 알려진다. 그룹의 크기는 각 멤버는 그룹의 크기만큼의 그레이 코드를 각자 계산하기 위하여 사용되기 때문에 정확한 숫자는 알 필요가 없다 그리고, 트리의 구성동안 새로이 추가 및 탈퇴되는 멤버에 대해서는 고려하지 않는다 각 멤버로부터 대역폭 정보의 수집은 bootstrap 메커니즘[20]을 통하여 각 코어가 갖고 있다 세션관리자는 코어의 대역폭 정보를 마찬가지의 방법으로 유지·관리한다

$$|G| = m \cdot Core_{\max} \quad \text{where} \quad Core_{\max} = max[Core_j]_{j \in SM} \tag{1}$$

여기서, |G|는 그룹의 크기이고 m은 한개의 코어가 관리하는 메버의 최대 개수이며 Corej는 세션관리자가 관리하고 있는 코어들의 식별값 i, SM은 세션관리자가 갖고 있는 코어의 식별값의 집합이다 트리 생성과 거의 동시에 메시지를 보냄을 알리는 제어메시지 'Data_REQ'를 전송함으로써 데이터 전송을 바로 시작하며, 'Data_REQ'메시지의 보장은 인터넷 인프라에서 제공하는 TCP프로토콜의 기능으로 해당 메시지의 전송을 보장한다. 트리 생성 도중에 실패하거나 탈퇴한 멤버가 존재한다면, 각 노드가 자신의 자식노드를 생성시 시작하는 타이머가 종료될때까지, 응답이 없으면 탈퇴로 간주한다 그러나, 이러한 경우는 드물게 존재함으로 가정한다.

각 노드는 자신이 데이터를 전달해야 하는 자식노드

입력 노드 i의 Gray Node 라벨 L, 남은 그룹크기 H, 소스 노드 S
출력 노드 i의 자식 노드 리스트와 라벨 I

```
// Perform MakeDeliveryTree until H=0
Calculate Gray Code with |G|
MakeDeliveryTree(L,H)
{
  N=H,
  for(j=0,j<nj++)          //비트스트링의 개수 n번 수행
  {
    I=FindChild(L),        //해당노드의 가능한 자식노드를 검색
    If(I && G-1(I)>G-1(L)||I==S)
    {
      N--,
      Request Cs to Addri,
      Store its child information,
      Send Data_REQ to I,
    }
  }
}
```

그림 1 트리 생성시 각 노드의 자식 검색 알고리즘

와 자신의 부모노드 정보만을 갖고 있다. 트리 생성 시 이용되는 식별자는 그레이(Gray) 코드[5]를 이용함으로써, 트리 생성 시간을 단축하고 코어를 둠으로써 각 호스트가 유지해야 하는 유지관리비용을 줄였다.

그룹에 참여하고자 하는 노드는 3.2에서 언급하고 있는 절차에 따라 그룹에 가입하여 메시를 형성하고, 자신의 코어로부터 자신의 식별값 i를 할당 받는다. 여기서, i값의 할당은 각 코어가 bootstrap메커니즘에 의해 획득한 각 노드의 대역폭 정보를 기반으로 내림차순으로 정리되어 '0'값에서부터 순차적으로 할당받으며 수식(2)와 (3)을 통해서 이루어진다. 이렇게 할당된 i값을 이용하여 그레이코드를 생성하며, 이렇게 생성된 코드를 이용하여 그림 1와 같은 알고리즘을 이용하여 트리를 생성한다.

트리를 생성하기 위해서, 데이터를 전송하고자 하는 소스는 자신의 자식노드를 검색한다. 그림 1의 트리생성 알고리즘에 따라 먼저 가능한 그레이코드를 생성하여 자식 노드 찾아낸 후, 자신의 코어에게 해당 노드의 식별자(예, 주소) 정보를 요청한다. 코어로부터 획득한 주소 정보를 가지고 데이터 전송을 알리는 'Data_REQ' 요청 메시지를 보낸다. 수신한 자식 노드들은 자신의 자식노드를 검색하여 같은 과정을 반복하여 트리를 구성해 내도록 한다. 그림 3에서 그룹의 크기가 5인 경우의 전송 트리 전송과정과 각 노드가 저장해야 할 정보를 보여주고 있다. 각 노드는 자신의 부모 노드와 하위 자식노드 정보만을 갖고 있다. 본 정보는 전송 중에만 저장되는 정보이며, 전송이 끝난 후에는 해당 정보도 함께 삭

```
입력: 그룹 멤버 i의 라벨 L
출력 : 가능한 i의 자식노드 라벨 Child
FindChild(L)
{
  Child=LnLn-1…Lk+1(1-Lk)…L2L1;
  return Child;
}
```

그림 2. 가능한 자식 라벨 검색 알고리즘

그림 2는 그림 1의 트리생성 알고리즘에서 사용되는 알고리즘으로 자신이 갖을 수 있는 가능한 자식노드의 그레이코드 라벨을 검색하는 과정이다. 그림 3은 제안한 One-Time트리 생성 알고리즘을 이용하여 그룹크기가 5일 때, 트리가 생성되는 과정을 보여준다. 본 과정은 순차적으로 예시한 것이며, 두번째 과정과 세번째 과정은 거의 동시에 자신의 자식노드를 찾게 된다. 먼저, 스트림을 보내고자 하는 소스는 그레이코드를 생성하여 자신이 갖을 수 있는 자식 노드를 검색하여 '001, 010'을 찾아내고 해당 노드의 식별자 정보 'R1, R2'를 저장한 후, R1과 R2에게 데이터 전송이 시작됨을 알리는 'Data_REQ'메시지를 전송한다. 해당 메시지를 전송 받은 R1과 R2는 각각 자신의 자식노드를 검색하고 2·3단계 과정을 수행함으로써 전송트리를 생성한다. 자신의 자식노드를 찾은 각 노드는 'Data_REQ'메시지를 전송 후, 바로 실시간 스트림을 전송하기 시작한다.

3.2 확장성을 제공하는 그룹관리

본 논문에서는 그룹관리를 위한 멤버간의 메시를 형성한다. 이렇게 형성된 메시는 그룹관리를 위해서 사용되며, 전송트리를 생성하는데 있어서의 기반구조가 되지는 않는다. 메시를 기반으로 하여 그룹의 가입 및 탈퇴 그리고, 호스트의 탈퇴 또는 고장으로 인한 네트워크의 분할의 복구를 수행한다. 표 2은 One-Time트리에서 사용되는 메시지를 정의한 것이다.

3.2.1 제어 위상의 형성

제어 위상는 단지 그룹관리를 위해 형성되는 메시이므로, 어떤한 제약도 존재하지 않으며 각 호스트는 유니캐스트 라우팅 알고리즘에 따라 연결된다. 각 메시에는 코어가 존재하며 각 코어들이 메시상의 멤버를 부분적으로 관리한다. 그룹관리를 분산 시킴으로써 그룹의 크기의 증가에도 크게 영향을 미치지 않도록 하였다.

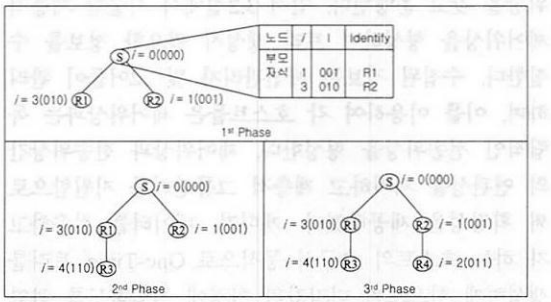


그림 3. 트리생성과정 (그룹크기=5)

각 코어는 자신의 전송률 이하의 멤버를 관리한다. 각 코어는 자신을 포함한 m개의 멤버리스트(Member List)를 가지며, 각 코어는 자신이 관리하는 멤버리스트의 최상위 i를 가진다. 따라서, 자신이 해당하는 코어에 따라서 순차적으로 i값이 할당되므로, i값의 할당에 큰 오버헤드를 갖지 않는다. i의 할당은 다음 수식 (2)과(3)의 규칙에 따라 할당되도록 한다. 여기서, 코어의 i값은 수식 (2)에서의 Corej이고, 각 멤버의 i

값은 수식 (3)에서의 Member(j)이다.

$$Core_j = mj\ ,\ \forall j \in Z^+ \tag{2}$$

$$\exists j, Member(j) = k$$
$$s.t.\ \ mj < k < mj + m,\ \ \forall k \in Z^+ \tag{3}$$

각 Corej는 수식 (2)와 같이, m의 배수의 형태로 i값을 전송률에 따른 내림차순으로 정렬되어 할당 받으며 이는 세션 메니져가 수행하도록 한다. 코어가 아닌 각 멤버들은 자신이 속한 코어로부터 i값을 마찬가지로 전송률에 따른 내림차순으로 정렬되어 할당받는다. 이때 각 멤버들은 주기적으로 자신의 가용 대역폭 정보를 주기적으로 코어에게 보내는 메시지에 함께 실어서 보내도록 한다. 각 코어는 이에 따라 라벨 재정렬을 수행하나, 전송 트리가 형성되어 데이터가 전송 중인 경우, 트리의 재설정 등은 수행하지 않는다. 변경된 정보는 다음 전송트리가 형성될 때에 새로이 적용된다. 마찬가지로, 각 코어는 센션관리자에게 보네는 주기적 메시지에 자신의 가용 대역폭 정보를 함께 보내도록 하며 세션관리자 역시 코어들의 라벨 재정렬을 수행한다.

이렇게 형성된 메시는 계층적 구조를 갖고 있으며, 그림 4와 같은 형태의 메시를 구성한다. 8명의 멤버가 존재할 때의 메시 구조를 보여주고 있으며, 해당 메시는 유니캐스트 라우팅 알고리즘에 따라 최단거리로 형성된다. 세션 메니져는 코어리스트(Core List)를 유지하고 있어, 가입요청시 해당하는 코어의 정보를 가입을 원하는 호스트에게 전달해주는 역할을 한다. 각 코어 A와 E는 자신이 관리하는 멤버리스트(Member List)의 정보와 다른 코어에 대한 정보(Neighbor Core List)를 갖고 있다. 코어 A, E간의

표 2. One-Time트리 메시지 정의

메시지	설명
Data_REQ	데이터 전송이 시작될 것을 알림
Join_REQ(ID,TR)	그룹에 가입을 요청
Join_ACK(ID)	가입이 완료되었음을 알림
Ctrl_Send(ID,TR)	코어에게 보내는 가입된 새로운 노드의 정보
REQ_ALIVE	자신의 생존여부를 알리기 위한 주기적인 제어메시지가 이웃으로부터 수신되지 않을때, 해당 멤버의 생존여부를 알기 위해 요청

메시와 하위 부분 적으로 같은 코어를 가진 노드들끼리의 메시가 형성된 구조를 볼 수 있다. 이렇게 형성된 제어 위상은 주기적인 메시지의 교환을 통해서 유지되며. 서로의 정보를 교환한다. 주기적인 메시지를 주고받는 대상은 그림 4에서는 {A,B,C,D}, {E,F,G,H}와 {A,E}이며, 세션 관리자와 코어간의 메시지 교환은 코어가 탈퇴하거나 중지되어서 새로운 코어가 선출되었거나, 새로운 코어가 생성된 경우에만 메시지를 교환한다.

3.2.2 가입 및 탈퇴 메커니즘

코어의 선택은 처음 그룹에 가입하는 호스트는 무조건적으로 코어로 지정되며, 해당 코어가 관리하는 노드의 수는 m개로 제한한다. 그 이상이 되었을 때에나, 전송률 이상의 호스트일 때에는 세션 관리자(Session Manager)에게 자신에게 가입시킬 수 없음을 알린다. 해당 세션 관리자는 전송률의 기반으로 새로이 코어를 지정하는 방식을 취하고 코어의 라벨 i값을 재정렬한다. 각 코어는 자신의 전송률 이하의 노드를 관리하도록 한다. 이는 세션 관리자가 가진 코어 리스트(Core List)의 정보를 이용하여 제한 가능하다.

공유 기반의 그룹에 가입시에는 그림 5와 같은 절차를 따른다. 여기서 세션관리자는 분산될 수 있으며, 그에 대한 제어기법은 다루지 않는다. 그룹에 가입하고자 원하는 호스트가 세션 관리자에게 가입요청(1)을 하면 세션 관리자는 코어가 존재하지 않거나 새로운 코어가 필요하면 단순히 어떠한 정보도 담기지 않은 단순한 가입수락(2.1) 메시지를 보내며, 코어가 존재하는 경우에는 그 호스트에게 코어 식별자 정보가 담긴 가입수락(2.2) 메시지를 전송한다. 이를 수신한 호스트는 마지막으로 자신의 코어에게 자신의 정보-식별자, 전송률-를 전송(2.2)해줌으로써 가입이 완료된다.

그룹에 가입한 멤버의 탈퇴는 자신이 속한 코어에

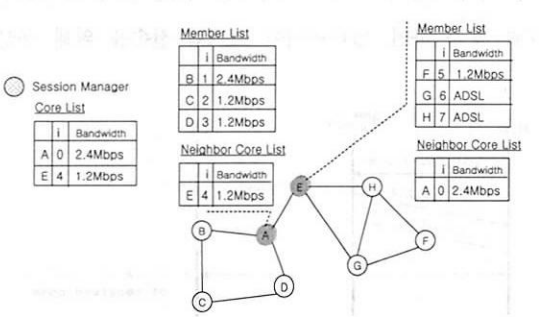


그림 4. 그룹관리를 위한 메시구조

게 탈퇴함을 알림으로써 이루어진다. 이러한 가입 또는 탈퇴 그리고 호스트 실패시는 메시의 재구성을 필요로 한다. 메시의 재구성 역시 확장성 있는 그룹관리를 위해서는 제어 위상의 재구성 시간 및 메시지 오버헤드가 크지 않아야 한다. 본 논문에서는 위에서 언급한 문제 외에 제어위상이 실제 데이터 전송을 위한 트리에 미치는 영향을 최소화함으로써, 그룹관리와 데이터 전송의 독립성을 강화하였다.

그룹의 가입시에는 자신이 속한 코어로부터 자신의 이웃 정보를 전송받아서, 자신과 가장 가까운 호스트로 자신의 이웃을 설정한다. 멤버의 탈퇴 및 실패시는 주기적으로 교환되는 메시지가 일정 시간 동안 오지 않으면 해당 멤버의 이웃들이 해당 멤버에 대해 'REQ_ALIVE'메시지를 보냄으로써 멤버의 삭제를 확인하여 멤버의 탈퇴 사실을 확인한다.

Ⅳ. 성능분석

본 논문에서는 생성된 One-Time트리에서 소스 노드의 최대 전송률 하에서 각 수신자의 최대 수신 대역폭을 보장하는 품질의 데이터를 수신할 수 있다는 것을 증명하며, 또한 전송 요구시에만 생성되는 One-Time트리의 생성시간이 길지 않음을 보이도록 한다. 그룹관리 측면에서는 확장성을 보이기 위해 멤버의 증가에 따른 제어 메시지의 증가도를 측정한다. 대역폭을 고려한 대표적인 응용멀티캐스트 기법인 Narada[8]와 데이터 수신품질 보장과 함께, 트리 생성을 위한 시간 복잡도 그리고 제어위상에서의 제어 메시지 교환비용을 비교분석하여 One-Time트리 생성을 위한 시간 복잡도가 크지 않고, 확장성 또한 우수함을 증명한다.

4.1 One-Time트리 생성 알고리즘

정리1. 송신자가 가진 전송률을 $t_{\max}$라 하자. 이때, 각 수신자들이 가진 대역폭을 $r_i, 1 \leq i \leq |G|$ $i \in N$라 하면, 멀티미디어 스트림 전송을 위해 구성된 One-Time 트리의 임의의 부모 노드 r_p와 r_c는 다음의 관계를 만족한다. 여기서 r_p와 r_c는 $r_p, r_c \in \{r_i\}_{i \leq |G|}$를 만족한다.

$$\begin{cases} r_c \leq r_p \leq t_{\max} & if \quad t_{\max} = \max\{t_{\max}, \max\{r_i\}_{i \leq |G|}\} \\ t_{\max} \leq r_c \quad or \quad t_{\max} \leq r_p & else \quad except \quad r_p \leq t_{\max} \leq r_c \end{cases}$$

증명. 소스노드의 전송률이 가장 높은 경우 즉, $t_{\max} = max\{t_{max}, max\{r_i\}_{i \leq |G|}\}$이면 노드 r_p가 자신의 자식노드를 검색시 그 자식 노드 r_c는 자신이 가진 i값 이상인 값을 취하게 되므로 당연하다. $t_{\max} = min\{t_{\max}, min\{r_i\}_{i \leq |G|}\}$인 경우에는 소스가 전송할 수 있는 최대 전송률이 각 노드가 가진 대역폭의 최소값을 취하므로 $t_{\max} \leq r_c or\ t_{\max} \leq r_p$인 관계를 만족한다. 그러나, One-Time트리 생성 조건에 의해 $r_p \leq t_{\max} \leq r_c$인 경우는 생성되지 않도록 하였기 때문에 정의에 의해 이는 존재하지 않는다.■

정리 1은 생성된 소스 기반의 트리가 소스의 최대 전송률을 넘지 않는 선에서 각 수신 노드의 수신환경을 고려하여 만족할 만한 전송품질을 제공할 수 있음을 증명한다. 여기에서, 데이터 전송 동안에는 가용대역폭의 변화를 반영하지 않는다. 본 One-Time트리 생성의 시간복잡도는 $O(NlogN)$이다. 자신의 노드를 검색하는 시간은 상수 시간 $O(1)$이고, $logN$만큼 수행한다. 이를 멤버의 수만큼 증가하게 되므로 $O(NlogN)$의 시간이 걸린다. Narada[8]의 경우는 트리는 기존의 DVMRP[21]등의 알고리즘을 사용하기 때문에서, DVMRP알고리즘을 사용하였을 때, 트리가 생성되기까지의 시간복잡도는 $O(NlogN)$이다. 그러나, One-Time기법은 완전한 트리를 형성하기 전에 부모와 자식관계만 맺어지면 바로 데이터의 전송을 시작

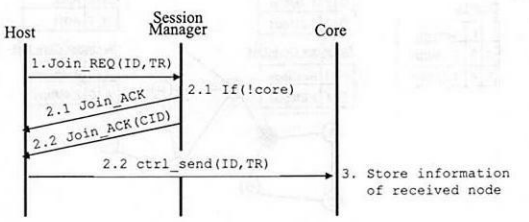


그림 5. 가입절차

표 3. One-Time 기법과 Narada의 트리생성시의 비교

	시간 복잡도	유지정보 크기	트리생성방법
One-Time	$O(N \cdot logN)$	$O(I)$	트리생성과 동시에 메시지 전송 시작
Narada	$O(N \log N)$	$O(m+I)$	수신자 기반의 트리생성 후 전송 시작

하게 되므로 최초의 데이터를 전송하기까지의 시간이 Narada[8]와 비교하여 첫 번째 데이터 전송까지의 시간은 짧으며, 자신의 부모노드와 자식노드 정보만을 가지므로 트리의 변경 또한 용이하다. 또한 제안한 One-Time은 소스로부터 동적으로 생성되어, 생성과 동시에 데이터를 전송하는 형태를 취하고, Narada[8]의 경우는 기존의 멀티캐스트 라우팅 알고리즘을 이용한 수신자 기반의 트리를 구성하기 때문에 그 트리생성부터 첫 데이터 전송까지의 시간 차이는 더 심하다. 또한, One-Time트리는 항상 유지 관리해야 하는 정보를 갖고 있지 않으며, 전송시에만 자신의 부모 및 자식에 관한 정보만을 유지하고 있으면 된다. 그러나, Narada[8]는 전체 멤버에 대한 정보와 트리가 생성되었을 때, 여러 그룹이 생성된 경우, One-Time기법에서의 유지해야 하는 정보를 함께 관리해야 한다. 즉, 자신이 메시지를 보내줄 자식이 속한 그룹 정보 등이다. 따라서, 다-대-다 환경에서의 요구에 따라 동적인 트리를 구성하는 One-Time이 트리를 생성하는 시간과 유지정보관리 측면에서 효과적이다. 표 3는 One-Time과 NARADA[8]의 트리 생성 측면에서 비교하였다.

Narada[8]의 경우, DVMRP 방식으로 트리를 생성시를 가정하였다. 그러나, 기존의 멀티캐스트 라우팅 알고리즘이 거의 수신자 기반의 트리생성 알고리즘을 사용하고 있기 때문에 마찬가지의 시간복잡도를 갖는다. 유지정보의 크기는 트리가 생성되거나, 전송이 시작되었을 때의 각 노드가 가져야 하는 유지정보를 말한다. 두 메커니즘 모두 자신이 데이터를 전달할 멤버에 대한 정보와 다자간의 통신을 지원하기 때문에, 속한 그룹에 대한 정보를 가지고 있어야 한다. 그 정보의 크기를 $O(l)$이라 하면, Narada의 경우는 그룹관리를 위한 $O(m)$의 정보를 함께 갖는다. 이는 비교를 용이하게 하기 위해서 위와 같이 표현한 것이며, Narada의 경우, 데이터를 보내야 할 자식들에 대한 그룹정보를 라우팅 테이블에 함께 가지고 있을 수도 있다. 그러나, 어떻게 관리를 하든 한 호스트가 관리해야 하는 유지정보의 크기는 $O(m+l)$이 된다. 또한 $m>>l$임은 당연하다.

4.2 그룹관리를 위한 제어위상

그룹관리의 확장성을 측정하기 위해, 그룹의 크기가 증가함에 따라 증가되는 제어메시지의 정도를 분석하며, 이를 위해 다음과 같은 가정을 한다.

- 세션메니져는 한개 이상이 될 수 있으나, 한개의 세션 메니져를 가정한다.
- 적어도 한개의 코어는 제어 위상에 존재한다.
- 코어는 갑자기 탈퇴하거나, 실패하지 않는다.
- 한개의 코어가 관리할 수 있는 호스트의 숫자는 m개로 제한한다.

표 4. 파라미터

파라미터	설명
$C_{One-Time}$	제안한 One-Time기법에서의 그룹관리를 위한 평균시간비용
C_g	기존 연구[1, 7]에서의 그룹관리를 위한 평균시간비용
$h_{joinReq}$	새로운 멤버의 수신 및 처리시간
$S_{joinAck}$	가입요청에 대한 응답 메시지 전송시간
$h_{ctrlSend}$	새로운 가입자에 대한 정보 송수신시간
$h_{periodic}$	제어위상의 유지를 위해 주기적으로 송신 또는 수신하는 시간
$h_{reqAlive}$	주기적인 메시지가 오지 않을때의 이웃노드의 상태를 점검하기 위해 메시지를 처리하고 송신하는 시간
m, R	하나의 코어가 관리하는 멤버의 수, 전체 그룹의 크기
p,q	가입률, 탈퇴 및 실패율

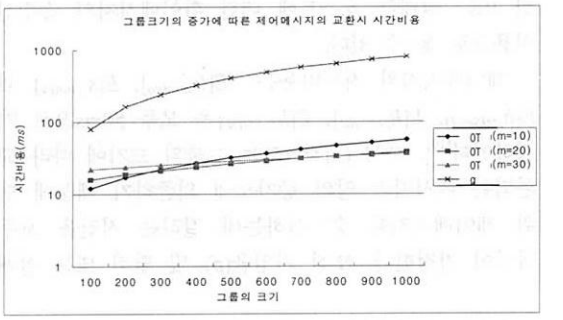


그림 6. 그룹크기의 증가에 따른 제어메시지의 교환시의 시간비용

- 각 제어 메시지의 송수신되고 처리되는 시간은 모두 일반적인 분포를 따르며, 모두 같은 분포를 따른다.

그룹의 시간비용은 가입률(p)로 그룹에 가입하는데 필요한 메시지의 처리 및 송수신 시간과 그룹유지를 위한 주기적인 메시지의 처리와 송수신 시간, 마지막으로 주기적인 메시지가 이웃 노드로부터 수신되지 않을때의 탈퇴율(q)로 확인메시지의 처리 및 송수신시

간으로서 정의한다.

비교하는 대상 Narada[8]는 코어가 존재하지 않으며, 모든 호스트는 제어 위상의 모든 호스트와 그룹관리를 위해 제어메시지를 교환한다. 각 호스트는 멤버들 간의 메시를 구성하며, 메시는 전체 멤버간의 주기적인 메시지 교환을 통해서 유지·관리된다.

제안하는 One-Time트리와 기존 연구의 제어메시지의 교환에 걸리는 평균 시간은 다음 수식은 각각 (4) 그리고 (5)와 같다. 이를 위한 파리미터의 정의는 표 4과 같다.

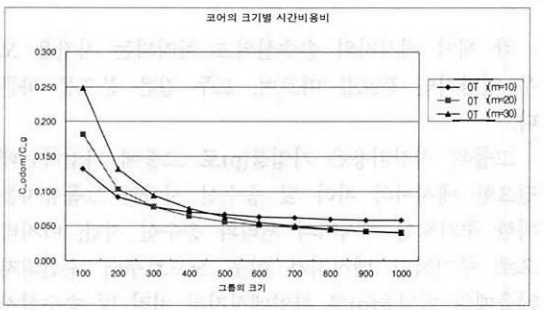
$$E[C_{ODOM}] = (E[h_{joinReq}] + E[s_{joinAck}] + E[h_{ctrlSend}]) + (m + \frac{R}{m})E[h_{periodic}] + mqE[h_{reqAlive}] \quad (4)$$

$$E[C_g] = (E[h_{joinReq}] + E[s_{joinAck}] + RE[h_{ctrSend}])p + RE[h_{periodic}] + RqE[h_{reqAlive}] \quad (5)$$

제안하는 One-Time트리에서의 그룹관리 기법에서는 가입 요청 메시지 그리고 그에 대한 응답, 가입하는 호스트의 정보를 송수신하는데 비용과 그룹을 유지하기 위한 주기적인 정보 송수신 비용 그리고, 탈퇴 또는 실패한 호스트에 대한 확인메시지의 송수신 비용으로 볼 수 있다.

제어메시지의 시간비용은 $E[h_{joinReq}]$, $E[s_{joinAck}]$ 과 $E[h_{ctrlSend}]$, $E[h_{periodic}]$, $E[h_{reqAlive}]$은 모두 50ms으로 가정하였다[19]. 수식 (4)과 (5)는 그룹의 크기에 따라 교환되는 메시지의 양의 증가도에 의존하기 때문에 단위 제어메시지를 송수신하는데 걸리는 시간을 모두 똑같이 가정한다. 또한 가입율(p) 및 탈퇴 또는 실패율(q)로 동일하게 0.3을 임의로 가정하였다.

그림 7. 코어의 크기에 따른 기존 연구 대비 시간비용 비율



이는 One-Time 기법(OT)이 각 m개를 관리하는 코어가 관리하기 때문에 기존의 연구(g)보다 상대적으로 더 그룹관리 비용이 그룹의 크기가 증가함에 따라 완만하게 증가함을 그림 6에서 보여주고 있다. 그림 6은 그룹의 크기가 100에서 1000까지 증가하며, m의 값은 10에서 30까지 증가한다. 기존의 연구는 그룹의 크기가 증가함에 따라 큰폭으로 증가함을 볼 수 있으며, 하나의 코어가 관리하는 멤버의 크기가 커짐에 따라 그 시간비용의 증가폭이 낮음을 알 수 있다. 또한 그룹의 코어의 크기가 30인 그룹관 코어의 크기가 20인 그룹을 비교하였을 때, 그룹의 크기가 100일 때는 코어의 크기가 30인 그룹이 제일 크나, 그룹의 크기가 증가하면서 코어의 크기가 20인 그룹과 30인 그룹이 비슷한 값으로 근사하는 것을 볼 수 있다. 따라서, 강건한 그룹관리를 위해서는 하나의 코어의 크기가 20정도가 적당하다.

그림 7는 기존 연구 [8]에서의 시간비용과 제안한 One-Time기법에서의 시간비용 비를 나타낸것으로, 그룹의 크기가 증가함에 따라, 제안한 기법의 그룹관리 성능이 더 좋음을 알 수 있다. 그리고, 한 코어 크기의 증가에 따라 그룹의 크기가 클수록 그룹관리의 확장성이 더 좋음을 알 수 있다. 하지만, 코어의 크기가 30으로 증가하였을때, 그 그룹의 크기가 100일때, 비용비가 0.25까지 증가함하였고, 그룹의 크기가 1000일때의 비용비는 코어의 크기가 20일때와 유사함을 알 수 있다.

V. 결 론

본 논문에서는 고용량의 멀티미디어 또는 스트리밍 데이터의 전송을 다양한 수신환경에서 전송가능한 One-Time트리 생성 알고리즘을 제안하였으며, 이를 위한 확장성 있는 그룹관리 기법을 함께 제안하였다. 성능분석을 통해, 트리생성 시간이 길지 않음을, 수신 대역폭을 보장함을 보였다. 또한 그룹의 크기 확장에 따른 제어 메시지의 증가도를 기존 오버레이 멀티캐스트 기법과 비교 분석하여 확장성을 만족함을 보였다.

본 논문은 차후, 실제 인터넷 상에서 구현할 예정이며, 구현을 통해서 본 기법이 실제 적용하였을 때도 효율적임을 보이고자 한다. 또한 다자간의 환경에 더욱 적합하고자, 송수신자간의 버퍼관리 기법에 대한 연구가 필요하다.

참 고 문 헌

[1] Y.-H. Chu, S G Rao, and H. Zhang. "A Case for End System Multicast," ACM SIGMETRICS, June 2000

[2] Paul Francis. "Yoid: Extending the Internet Multicast Architecture," http://www.icir.org/yoid/docs/ycHtmlL/ht-mlRoot.html, April, 2000

[3] S. Banerjee, B. Bhattacharjee, C. Commareddy. "Scalable Application Layer Multicast," ACM SIGCOMM, Pittsburgh, PA August 2002

[4] C. Semeria "Introduction to IP Multicast Routing," Internet Engineering Task Force, January 1997. Internet Draft, expires July 1997.

[5] J Liebeherr and B.S.Sethi "A Scalable Control Topology for Multicast Communications," IEEE infocom 98, March 1998.

[6] J. Jannotti, D. K Gifford, K. L Johnson, M F Kaashoek, and Jr. J W. O'Toole "Overcast: Reliable Multicasting with an Overlay Network," 2000

[7] M Castro, P. Druschel, A -M. Kermarrec, and A. Rowstron "SCRIBE A Large-Scale and Decentralized Application-Level Multicast Infrastructure," IEEE J. Selected Areas in Communication., 2002.

[8] Y Chu, S. Rao, S. Seshan, and H. Zhang "Enabling Conferencing Applications on the Internet using an Overlay Multicast Architecture," ACM SIGCOMM, 2001.

[9] S Saroiu, P. K Gummadi, and S D Gribble "A measurement study of peer-to-peer file sharing systems," MMCN, January 2002.

[10] J. Ledlie, J. Taylor, L. Serban, and M. Seltzer. "Self-organization in peer-to-peer systems," ACM SIGOPS European Workshop, September 2002

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

[12] T Nguyen and A. Zakhor "Distributed video streaming over internet In the SPIE Conference on Multimedia Computing and Networking," San Jose, California, January 2002.

[13] McCanne, S V. Jacobson, M. vetterli. "Receiver-driven Layered Multicast," SIGCOMM'96, pp117-130, 1996.

[14] John Apostolopoulos, Tina Wong, Wai-tian, Susie Wee "On Multiple Description Streaming with Content Delivery Networks," IEEE INFOCOM, June 2002.

[15] 피영수, 김기영, 윤미연, 신용태. "모바일 네트워크에서 MDC를 이용한 멀티미디어 지원방안," JCCI21, 4 30-5.2 2003

[16] 피영수, 윤미연, 김대원, 신용태. "이종환경에서 실시간 스트리밍 서비스를 위한 다대다 오버레이 멀티캐스트 기법," 춘계학술대회, 정보과학회, 2003

[17] B Zhang, S Jamin, and L. Zhang. "Host Multicast: A framework for delivering multicast to end users," In Proc. of IEEE INFOCOM, New York, NY, June 2002.

[18] ROWSTRON, A, AND DRUSCHEL, P "Pastry.Scalable, distributed object location and routing for large-scale peer-to-peer systems," http //research microsoft com/ antr/PAST/, 2001.

[19] S Pingali, D Towsley, J Kurose. "A Comparison of Sender-Initiated and Receiver-Initiated Reliable Multicast Protocols," In Proc INFOCOM, San Francisco, CA, October 1993

[20] Bill Croft, John Gilmore, "Bootstrap Protocol," RFC 0951, IETF, September 1985.

[21] D Waitzman, C. Partridge, "Distance Vector Multicast Routing Procotol," RFC 1075, IETF, November 1988.

윤 미 연(Mi-youn Yoon) 정회원

2000년 2월 : 가톨릭대학교 수학/컴퓨터학과 졸업
2002년 2월 : 숭실대학교 컴퓨터학과 석사
2004년 2월 : 숭실대학교 컴퓨터학과 박사과정 수료
2004년 3월~현재 : 숭실대학교 컴퓨터학과 박사과정

<관심분야> 멀티캐스트, 센서네트워크, ad-hoc network, 정보보호, wireless 네트워크, IPv6

김 기 영(Ki-young Kim) 정회원

1996년 2월 : 상지대학교 전자계산학과 졸업
1995년~1997년 : 삼보정보통신기술 연구소 연구원
1999년 : 숭실대학교 컴퓨터학과 석사
2003년 : 숭실대학교 컴퓨터학과 공학박사
2004년~현재 : 서일대학 정보통신계열 소프트웨어 전임강사

<관심분야> 멀티캐스트, Mobile-IP, 실시간 프로토콜, 정보보호

김 대 원(Dae-won Kim) 정회원

2002년 : 영동대학교 컴퓨터공학과 졸업
2004년 : 숭실대학교 컴퓨터학과 석사
2005년 3월~ : 숭실대학교 컴퓨터학과 박사과정

<관심분야> 멀티캐스트, DRM, 정보보호, IPv6

신 용 태(Yong-tae Shin) 정회원

1985년 : 한양대학교 산업공학과 졸업
1990년 : Iowa대학교 전자계산학과 석사
1994년 : Iowa대학교 전자계산학과 박사
1995년~현재 : 숭실대학교 정보과학대학 컴퓨터학부 부교수

<관심분야> 멀티캐스트, 정보보호, DRM, 그룹통신