오버레이 멀티캐스트 트리 최적화를 위한 동적 분산화 방법

최 한, 김종권
서울대학교 컴퓨터공학부
hchoe@popeye.snu.ac.kr

A Dynamic Distributed Mechanism for Optimization of Overlay Multicast Tree

Han Choe and Chong-kwon Kim
Information Networking and Computing Lab.
Department of Computer Science, Seoul National University.

요 약

오버레이 멀티캐스트의 성능 향상을 위해서는 멀티캐스트 트리의 최적화가 필요하다. 오버레이 멀티캐스트 트리의 최적화 문제는 그래프 상에서 각 노드의 차수가 제한되어 있을 때 지름이 가장 작은 트리를 만드는 문제와 같으며, 이는 NP-complete문제로 알려져 있다. 본 논문에서는 멀티캐스트 트리의 최적화를 위한 동적, 분산화 알고리즘을 제안한다. 각 사용자는 자신에서 출발하는 최대 경로의 길이를 계산하고, 주변 사용자와 변을 교환함을 통해 멀티캐스트 트리의 지름을 줄여나가게 된다. 이 알고리즘은 계산 상의 부하가 적고, 동적인 환경에 적응할 수 있으며, 중앙의 통제가 필요하지 않다는 장점을 가지고 있다.

1. 서 론

인터넷 방송, 화상 회의 등의 집단 간의 통신을 위해서는 효율적이고 확장 가능한 멀티캐스트(multicast) 메커니즘이 필요하다. 수년 전까지 IP 멀티캐스트가 이를 위한 적절한 메커니즘으로 여겨져 왔다. 그러나 IP 멀티캐스트는 라우터의 구현, 혼잡 제어와 신뢰성 있는 전송에서 여러 가지 문제점에 맞닥뜨리게 되었으며, 그에 대한 대안으로 오버레이(overlay) 멀티캐스트 방법이 제안 되었다.

IP 멀티캐스트에서 패킷의 복제와 전송을 라우터가 담당하는 반면, 오버레이 멀티캐스트에서는 단말의 사용자가 패킷의 복제와 전송을 담당하게 된다. 즉 단말 사용자들로 이루어진 응용 계층 상에서의 멀티캐스트 트리를 구성하여 그 트리의 경로를 통해 패킷을 전송한다. 각 사용자는 패킷을 받은 후에 트리 상의 다른 이웃 단말 사용자에게 패킷을 전송하게 된다. IP 멀티캐스트는 구현을 위해 라우터의 개선이 필요한 것에 반해, 오버레이 멀티캐스트는 각 사용자의 응용 프로그램만 갖추어지면 구현될 수 있다는 장점이 있다.

반면 오버레이 멀티캐스트는 IP 멀티캐스트에 비해 지연 시간이나 대역 폭 사용의 측면에 있어서 비효율적이라는 단점이 있다. 오버레이 멀티캐스트에서는 최대한 이런 비효율성을 줄이는 멀티캐스트 트리를 구성하는 것이 중요한 목표이다.

화상 회의, 인터넷 방송 등의 실시간 응용 시스템을 위한 오버레이 멀티캐스트 트리가 충족 시켜야 하는 조건은 크게 두 가지이다. 첫 번째로, 사용자, 즉 트리 상의 노드의 차수가 적절한 상한 이하여야 한다. 트리 상의 특정 노드의 차수가 지나치게 높다는 것은, 그 사용자가 패킷을 받았을 때 복제해서 보내 주어야 할 이웃 사용자가 그만큼 많다는 것을 의미한다. 따라서 그 사용자가 패킷을 복사하고 처리하는 프로세싱의 과정에서 오버헤드가 커지고, 또한 그 사용자가 접속하고 있는 네트워크 상의 링크로 동시에 많은 데이터를 보내 주어야 하기 때문에 대역 폭의 한계로 인해 각각의 이웃 사용자에 대한 전송 속도가 떨어지게 된다. 두 번째 조건은, 멀티캐스트 트리의 지름(diameter), 즉 트리 상의 경로 거리로 볼 때 가장 먼 두 사용자 간의 거리가 작아야 한다는 것이다. 두 사용자 간의 트리 상의 경로 거리가 크면 두 사용자간에 데이터를 보낼 때의 지연 시간이 커지기 때문에, 화상 회의와 같은 응용 시스템에 적합하지 않게 된다.

각 노드의 차수가 상한을 가지고 있을 때 지름이 최소인 트리를 구성하는 문제는 NP-complete문제임이 알려져 있다. 지금까지 이 문제를 위한 여러 가지 heuristic들이 제안되어 있지만, 아직 동적인 환경에 대응할 수 있는 분산화 알고리즘은 제시되어 있지 않다. 본 논문에서는 오버레이 멀티캐스트 트리 최적화를 위한 동적 분산화 알고리즘을 제시하고자 한다.

2장에서는 지금까지 제시된 방법들을 살펴보고, 3장에서는 새로운 알고리즘을 제시하고자 한다.

2. 관련 연구 결과

[2] 에서는 각 노드의 차수가 상한을 가지고 있을 때 지름이 작은 트리를 구성하기 위해 Prim의 minimum spanning tree 알고리즘을 변형한 알고리즘을 제시하였다. 하지만 이는 최적 값을 보장하지 못하며, 사용자가 가입하고 탈퇴하는 동적인 상황에 대처할 수 없다.

[2], [3]는 지름이 일정한 상한 이하가 되면서 각 노드의 차수가 가능한 한 고르게 분포하는 트리를 구성하는 알고리즘을 제시하였다. 그러나 이 알고리즘은 계산 복잡도가 크고 동적인 환경에 대응하지 못한다.
[4]에서는 한 곳에서만 데이터를 전송하는 경우에 트리를 최적화하는 동적인 알고리즘을 제시하였다. 이 경우에는 트리의 지름이 아닌, 전송자로부터의 최대(평균)거리를 최소화하는 방법을 제시하였다.

3. 동적 분산화 트리 최적화 방법

3.1 시스템 모델

그래프를 사용하여 시스템을 표현한다. N명의 멀티캐스트 사용자가 있을 때 i 번째 사용자를 M_i라고 부른다. E_{ij}는 M_i 와 M_j간을 연결하는 응용계층상의 선을, C_{ij}는 M_i 와 M_j간의 네트워크상의 경로 거리(혹은 지연 시간)를 의미한다. $\{M_i\}$를 정의 집합으로 $\{E_{ij}\}$의 부분집합을 변의 집합으로 갖는 트리 T를 오버레이 멀티캐스트 트리라고 부른다. OC_{ij}는 T상에서의 M_i와 M_j간의 거리를 의미한다. T상에서의 M_i의 차수는 Deg_i로 나타내고, 차수의 상한은 $Degbound_i$이다. 그림 1 은 한 네트워크 모델을 나타내고 있다. 왼쪽 그림은 M_1이 전송할 때, 응용 계층 상에서의 멀티캐스트 트리를 나타내고, 오른쪽 그림은 실제 네트워크 상으로 패킷이 전송되는 경로를 나타낸다. 여기서 $OC_{13} = C_{13} = 3 + 2 + 3 = 8$ 이다. $C_{14} = 3 + 2 + 3 = 8$ 이고, $OC_{14} = C_{12} + C_{24} = 6 + 8 = 14$ 이다. $Deg_1 = Deg_2 = 2$ 이고, $Deg_3 = Deg_4 = 1$ 이다.

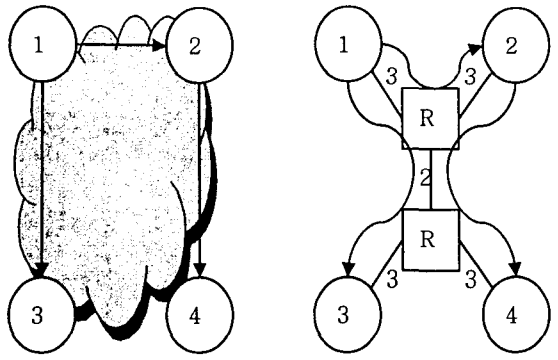


그림 1. 오버레이 멀티캐스트 트리의 한 예와 패킷의 전송 경로. 사용자 2는 사용자 1에게서 받은 패킷을 복제하여 사용자 4 에게 전송한다. R은 라우터를 나타냄.

3.2 각 사용자가 가져야 할 정보

사용자 M_i 는 오버레이 멀티케스트 트리 T상에서 이웃한 사용자들을 알고 있다. 이웃 사용자 M_j에 대해서 경로 거리 C_{ij}를 알고 있고(이 경우 $C_{ij} = OC_{ij}$임), M_i 에서 시작하며 E_{ij}를 포함하는 T상의 경로(path)들 중에서 가장 긴 경로의 길이를 알고 있다. 이를 $MaxOC_{ij}$로 나타낸다. 그림 2 에서 $MaxOC_{43} = OC_{41} = 6 + 9 = 15$ 이다.

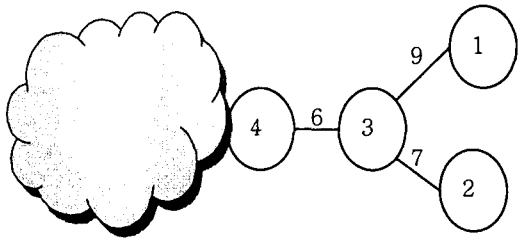


그림 2. $MaxOC_{31}$=9, $MaxOC_{32}$=7, $MaxOC_{43}$=15.

3.3 $MaxOC_{ij}$ 계산

$MaxOC_{ij}$는 T상의 리프 노드(leaf node, 차수가 1인 노드)들로부터의 플러딩(flooding)을 통해 계산된다. 그림 3 에서 M_5가 M_4에게 자신이 리프 노드임을 알려주면 M_4는 이를 통해 $MaxOC_{45} = C_{45} = 7$ 임을 알게 되고, 이 값을 M_3 에게 알려준다. M_3 은 이를 통해 $MaxOC_{34} = MaxOC_{45} + C_{34} = 7 + 9 = 16$ 임을 알게 된다. 역시 M_3은 이 값을 M_1, M_2 에게 보내주게 되고, 이를 통해 M_1 은 $MaxOC_{13} = 22$ 임을, M_2 는 $MaxOC_{23} = 23$ 임을 계산하고 역시 다른 이웃 사용자에게 값을 전달해 주게 된다. 이런 식으로, 어떤 사용자가 처음으로 MaxOC값을 전달 받거나, 현재 가지고 있는 값보다 큰 MaxOC값을 받게 되면 자신의 MaxOC값을 바꾸게 된다. 모든 플러딩이 완료된 후에는 모든 사용자가 자신의 모든 이웃에 대한 MaxOC 값을 정확히 알게 된다.

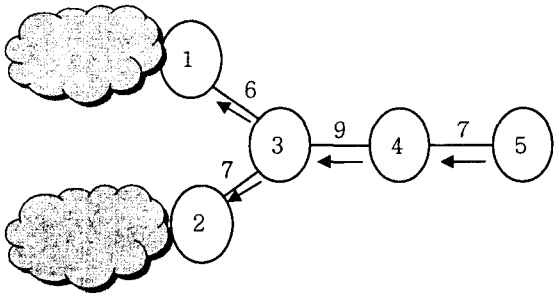
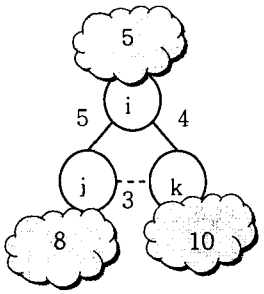


그림 3. 플러딩을 통한 MaxOC 계산 과정.

3.4 동적 트리 최적화의 과정과 그 특징

동적 트리 최적화는 세 사용자 사이에서 이루어지게 된다. M_i는 자신의 MaxOC 값들 중에 가장 큰 두 값을 선택하게 된다. 이것을 $MaxOC_{ij}$, $MaxOC_{ik}$ 라고 하자. M_i는 M_j, M_k에게 트리 최적화 과정을 시작할 것을 알리게 되고, M_j, M_k 는 각자의 MaxOC 값과 측정을 통해 알 수 있는 C_{jk}를 M_i에게 알려준다. 이를 기초로 M_i는 세 사용자 간의 변을 교환했을 때 M_i, M_j 혹은 M_k를 통과하는 트리 상의 최대 경로의 길이가 얼마인지 계산하게 된다. 세 사용자의 차수가 상한을 넘지 않는 범위 내에서 최대 경로의 길이를 최소화 하는 방향으로 변을 교환하게 된다. (그림 4)
변을 교환하기 전에 그래프 상에 회로가 존재하지 않았다면 변을 교환해도 이 그래프에 회로(cycle)가 생기지



Max {

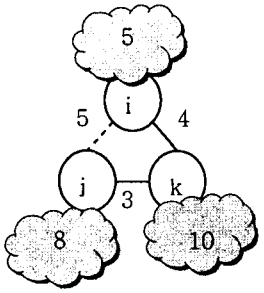
$MaxOC_{j?}+C_{ji}+C_{ik}+MaxOC_{k?}$

,

$MaxOC_{i?}+C_{ij}+MaxOC_{j?}$,

$MaxOC_{i?}+C_{ik}+MaxOC_{k?}$,

} = 8 + 5 + 4 + 10 = 27



Max {

$MaxOC_{j?}+C_{jk}+C_{ki}+MaxOC_{i?}$,

$MaxOC_{i?}+C_{ik}+MaxOC_{k?}$,

$MaxOC_{j?}+C_{jk}+MaxOC_{k?}$,

} = 8 + 3 + 10 = 21

그림 4. 변 교환의 예. C_{ij}=5, C_{jk}=3, C_{ki}=4, $MaxOC_{i?}$=5, $MaxOC_{j?}$=8, $MaxOC_{k?}$=10.

않는다. 그림 4 에서 변 교환 후에 회로가 생길 수 있는 가능성은 E_{jk}를 포함하는 회로의 경우 밖에 없다. 이는 E_{jk} 외에도 M_j 와 M_k 를 연결하는 다른 경로가 있다는 것을 의미하는데, E_{ij}가 트리에서 제외되었으므로 이는 불가능하다.

변을 교환한 후 M_i는 변화된 상황을 기초로 자신의 정보를 변경하게 된다. 그림 4 의 경우에는 M_j를 이웃에서 제외시키고, $MaxOC_{ik}$ 를 기존 값인 10 에서 C_{kj} + $MaxOC_{j?}$ = 11로 변경하게 된다. 이 MaxOC 정보를 M_i에서 시작하여 플러딩함을 통해 사용자 전체가 변 교환에 의해 바뀐 MaxOC정보를 갖게 된다.

트리 상의 모든 사용자가 자신을 중심으로 한 동적 변 교환을 시도하게 된다. 만일 함께 동적 변 교환을 수행하고자 하는 사용자가 이미 다른 변 교환에 참여 중이라면, 잠시 대기하였다가 다시 변 교환을 시도하게 된다.

이런 트리 최적화 과정은 동시에 여러 곳에서 이루어지게 된다. 트리의 지름에 해당하는 경로 상에서 일어나는 변 교환은 모두 트리의 지름을 감소시키는 방향으로 변을 교환하게 되고, 변 교환 후에도 이전의 지름보다 길이가 긴 새로운 경로는 생기지 않는다. 트리의 지름에 해당하는 경로 밖에서 일어나는 변 교환은 트리의 지름에 직접적으로 영향을 미치지는 않게 된다. 그러므로 동시에 여러 곳에서 변 교환이 일어나더라도 트리의 지름은 증가하지 않는다. 모든 변 교환과 MaxOC의 플러딩이 완료된 후에는 모든 사용자가 올바른 MaxOC값을 갖게 된다.

이 동적 변 교환을 통해서 Deg가 Degbound를 넘지 않는 한도 내에서 트리의 지름이 감소하게 된다. Deg가 지나치게 커지지 않으므로 각 사용자의 프로세싱 오버헤드가 지나치게 커지지 않고, 또 사용자 주변 링크에 대한 대역폭 낭비가 생기지 않게 된다. 또한 트리의 지름이 감소하기 때문에 두 사용자 간의 전송의 최대 지연 시간이 감소하게 되어 실시간 응용에 적절한 환경을 만들게 된다.

이 방법은 중앙 집중적인 제어가 필요하지 않고, 짧은 시간 내에 수행할 수 있기 때문에, 사용자가 가입, 탈퇴하고, 네트워크 상의 지연시간, 대역 폭이 변화하는 동적인 환경에 잘 대처할 수 있을 것이라고 예상된다.

4. 결론

오버레이 멀티캐스트의 성능 향상을 위해서는 멀티캐스트 트리의 최적화가 필수적이다. 지연시간과 프로세싱 오버헤드를 줄이고, 전송 속도를 증가시키기 위해서는 각 점의 차수의 상한이 있고, 지름이 최소인 트리를 구성해야 한다.

현재까지 제시된 알고리즘은 최적해에 접근하지 못하고, 동적인 환경에 대처하지 못하며, 중앙의 관리자가 있어야 한다는 단점이 있다.

이 논문에서는 동적 분산화 트리 최적화 방법을 제시하였다. 각 사용자가 자신에게서 시작하는 트리 상의 최대 경로의 길이를 계산하고, 세 사용자 사이에서 변을 교환함을 통해 트리의 지름을 줄여나가게 된다. 변 교환은 회로를 만들지 않으며, 동시에 여러 곳에서 이루어질 수 있다.

이 방법은 크지 않은 제어 부하를 가지며, 동적인 환경에 대처할 수 있고, 분산된 사용자들이 직접 트리를 최적화할 수 있다는 장점이 있다.

5. 참고 문헌

[1] Yang-hua Chu, Sanjay G. Rao, and Hui Zhang. " A Case for End System Multicast." ACM SIGMETRICS 2000.

[2] Sherlia Y. Shi, Jonathan S. Turner, Marcel Waldvogel, " Dimensioning Server Access Bandwidth and Multicast Routing in Overlay Networks."

[3] S. Shi and J. Turner, " Routing in overlay multicast networks." in Proceedings of Infocom, June 2002.

[4] Suman Banerjee, Christopher Kommareddy , Koushik Kar , Samrat Bhattacharjee, Samir Khuller, " Construction of an Efficient Overlay Multicast Infrastructure for Real-time Applications." , Infocom' 03