

RED알고리즘을 이용한 멀티캐스트 라우터 기반 버퍼 관리에 관한 연구

최원혁^o, 김정선

rbooo^o@korea.com, jskim@hangkong.ac.kr

한국항공대학교 전자공학과

WonHyuck Choi^o, Jungsun Kim

Korea, School of Electronics, Telecommunication and Computer Engineering,
Hankuk Aviation University

요 약

이중 공유기반 트리 방식인 CBT 방식은 송신자와 수신자 모두 센터 노드(Core)에 이르는 최단경로를 통해 데이터를 주고받는 방식이다. 이 방식은 Core를 이용하여 멀티캐스트의 한계인 확장성문제를 해결하기 위해서 사용된 방식이다. 그러나 현재 멀티미디어 라우팅방식은 Best-Effort 방식의 패킷 스위칭은 가능한 한 패킷을 전송하려 하기 때문에 Core와 RP를 중심으로 데이터의 폭주 현상을 유발한다.

우리는 이와 같은 문제를 해결하기 위해서 RED 알고리즘을 이용한 Anycast 방식을 제안한다. Anycast 프로토콜은 Anycast 주소를 가진 수신자 그룹 멤버들 중, 가장 근접한 라우터나 지역의 Anycast 그룹 멤버들에게 패킷을 재분배할 수 있는 최적의 서버 또는 호스트에게 패킷을 전달하는 방식이다. 이와 같이 Anycast는 Core의 패킷을 재분배하여 근접한 라우터로 재분배 하여 최적의 호스트에게 전달할 수 있다. Core의 패킷을 재분배하기 위해서는 RED 알고리즘을 도입하여 Core의 트래픽 증가를 분산시킨다.

1. 서 론

오늘날 인터넷의 급격한 발전으로 인해 다양한 트래픽과 서비스가 생겨났다. 또한 데이터 전송기술이 고속화로 멀티미디어를 이용한 서비스에 대하여 요구가 급속히 증가하고 있다. 멀티미디어 서비스는 기존의 텍스트를 벗어나 음성, 오디오, 이미지등의 통합적 환경으로 제공 되고 있다. 이에 따라 워크스테이션뿐만 아니라 개인용 컴퓨터에서도 처리해야 할 데이터가 증가하고 있다. 또한 전송되는 응용들에서 각종 데이터를 이용하는 멀티미디어 통신이 급격한 일반화 양상을 보이고 있다.

현재 멀티미디어 라우팅방식은 Best-Effort 방식의 패킷 스위칭은 가능한 하나의 패킷을 전송하려 하기 때문에 Core와 RP를 중심으로 데이터의 폭주 현상을 유발한다. 이러한 혼잡을 제어하기 위해서는 네트워크의 혼잡 상태를 송신측, 주위의 라우터(B-core)에 알려주는 피드백이 필요하다.

송신측은 이러한 혼잡 신호를 받고 Core라우터로 내보내는 패킷을 줄여야 한다. 더불어 코어 주변의 다른 라우터는 데이터가 Core 주변으로 접근 되므로 Core주변의 다른 라우터는 Poor 현상이 발생한다. 그림1,2는 CBT 문제는 Core에 데이터 집중과 Core와의 다른 라우터는 Poor Core 현상을 보여준다. 이와 같은 것을 방지하기 위해서 Core 라우터가 아닌 다른 쪽의 B-core 라우터를 찾아서 패킷을 분산 전송해야한다.

우리는 이와 같은 문제를 해결하기 위해서 RED 알고리즘을

이용한 Anycast 방식을 제안한다. RED 알고리즘은 버퍼에 평균 큐의 크기를 이용하여 네트워크 혼잡상황을 미리 감지하고 제어하는 기능이다.

Anycast 프로토콜은 Anycast 주소를 가진 수신자 그룹 멤버들 중, 가장 근접한 라우터나 지역의 Anycast 그룹 멤버들에게 패킷을 재분배할 수 있는 최적의 서버 또는 호스트에게 패킷을 전달하는 방식이다. 이와 같이 Anycast는 Core의 패킷을 재분배하여 근접한 라우터로 재분배 하여 최적의 호스트에게 전달할 수 있다.

Core의 패킷을 재분배하기 위해서는 RED 알고리즘을 도입하여 Core의 트래픽 증가를 분산시킨다.

본 논문의 실험을 위해서 전통적인 멀티캐스트 프로토콜인 CBT방식과 RED 알고리즘을 이용한 패킷분산 Anycast방식을 시뮬레이션 상에서 구현하고 성능을 평가한다. 한편, 이후 본 논문의 CBT 프로토콜과 RED 패킷버퍼관리 알고리즘을 설명한다. 3장에서는 2장에서 제안한 방식에 대한 시뮬레이션 결과와 성능을 분석하고 4장에서 결론을 내린다.

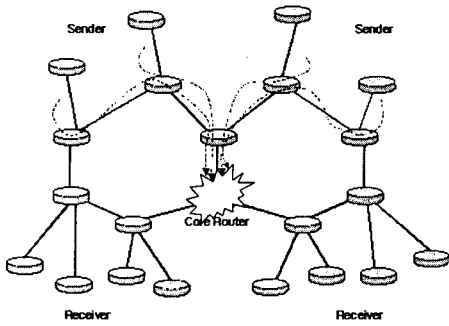


그림1 Bottleneck Core 현상

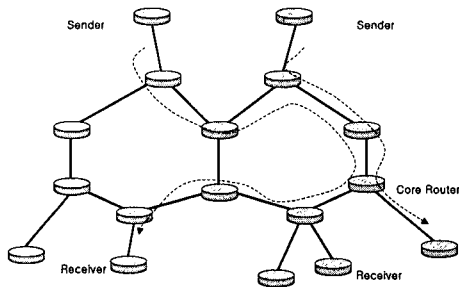


그림2 Poor Core 현상

2. RED알고리즘을 이용한 Anycast 분산 방식

본 논문에서는 다음의 제안된 네트워크 토폴로지를 이용하여 그룹의 멤버들을 형성 가정한다. 주어진 노드들의 그룹형성을 토대로 데이터 패킷을 전송시킨다. RED알고리즘을 이용해서 코어의 임계치보다 작은 경우는 CBT형식의 Core 형식을 선택하고, 코어의 임계치보다 증가된 경우는 Anycast방식을 도입한다.

또한 CBT는 라우터의 일정시간 RTT동안 주위의 라우터를 플러딩하여 송신자들에게 알려주므로 해서 송신자는 다른 라우터의 최단거리로 쉽게 접근할 수 있다.

2.1 RED를 이용한 anycast

기존의 CBT 방식은 단지 멀티캐스트 서비스를 직접 관련되는 라우터 혹은 네트워크들만을 포함하는 공유 멀티캐스트 방식을 유지하기 위해 제안된 방법으로 Core라우터에 트래픽을 집중시켰다. 이를 해결하기 위해서 RED 알고리즘을 도입하였다. CBT는 호스트가 IGMP의 호스트 멤버십에 리포트를 전송후 그룹에 참여한다. 이 메시지를 받은 지역 라우터는 join_request 메시지를 Core 라우터로 전송한다. 참여 메시지는 Core 라우터에 의해서 승인되고 멤버 승인과정이 끝나면 송, 수신은 Core라우터로부터 이뤄진다.

제안된 방식은 CBT의 Core에 RED 알고리즘을 도입하였다. CBT의 Core 라우터의 트래픽은 표2의 매개변수 권고치에 의해서 제어된다.

다음은 RED 알고리즘을 이용한 트래픽 제어 알고리즘이다. 위와 같이 RED 알고리즘은 라우터 코어의 평균 패킷의길이 일정 수준으로 넘어가면 라우터 코어의 전송되는 패킷을 미리 예측하는 기법으로 Core라우터의 버퍼의 트래픽 증가를 예측할 수 있다.

만약 avg가 max_{th} 보다 크면 Core의 트래픽 증가 되므로 이때 코어는 송신자에게 nonqu 메시지를 전송한다. nonqu 메시지를 받은 송신자는 Anycast 방식으로 전환된다. Anycast 프로토콜은 Anycast 주소를 가진 수신자 그룹 멤버들 중, 가장 근접한 라우터나 지역의 Anycast 그룹 멤버들에게 패킷을 재분배할 수 있는 최적의 서버 또는 호스트에게 패킷을 전달하는 형식으로 동작한다. 이때 사용하는 Anycast 주소는 동일한 서비스를 제공하는 서버들의 그룹을 정의하는데 사용된다. Anycast를 기존의 통신 방식들과 비교하여 일반적으로 수신자에 의한 네트워크 서비스로 구분하는 가장 큰 이유는, 임의의 정보에 대하여 서비스를 받고자 하는 수신자는 Anycast 주소를 통하여 지정된 다른 수신자들과 같은 특성을 공유하기 때문이다. 이러한 Anycast 수신자들과 연관된 송신자는 데이터 패킷을 Anycast 주소로 전송하고 라우터는 송신자로부터 가장 근접한 수신자에게 패킷을 전송한다. 송신자로부터 가장 근접한 수신자의 선택은 거리를 측정하는 특정한 라우터에서 담당하게 된다. Anycast 라우팅 프로토콜의 내부 속성상 유니캐스트의 특성을 멀티캐스트로 확장하였기 때문에 프로토콜의 동작 과정에서 각각의 라우팅 방식들은 내부적으로 상호 통신 작용이 이루어진다. Anycast 데이터의 송신자와 수신자 사이에서 이루어지는 프로토콜의 상호 통신 작용은 출발지(Source)에서 목적지(Destination) 방향 향한다. 다음은 CBT가 anycast 방식으로 전환되는 순서도이다.

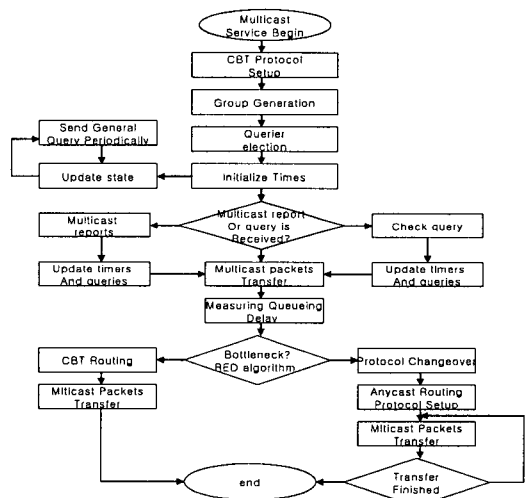


그림3.CBT가 anycast 방식으로 전환되는 순서도

제안한 RED 알고리즘을 이용한 Anycast 방식은 위와 같이 CBT에서 집중되는 트래픽 현상을 라우터의 버퍼를 예측하여 Anycast로 전환함으로써 멀티미디어와 같은 실시간을 요구하

는 데이터를 분산 송할 수 있다. 또한 서론에서의 Poor현상도 적절한 버퍼관리를 통한 Anycast로 운영됨으로 방지 할 수 있다.

3시뮬레이션

본 논문에서는 RED 알고리즘을 이용한 Anycast 라우터의 성능 평가를 기존의 멀티캐스트 기반의 CBT와 성능을 비교 분석하였다. 실험에서 송신자 수와 그룹 수를 가변 시키기 위해 10개의 송신자와 6의 그룹을 두었고, 각각의 그룹은 최대 3개의 수신자를 두어 그룹 멤버

그림3은 CBT가 anycast 방식으로 전환되는 순서도 수의 변화에 대응하도록 하였다. 각각의 멀티캐스트 링크는 1.5Mbps의 고정 대역폭과 10ms의 전파지연을 갖는다. 시뮬레이션은 10분 각 소스 특성에 따른 라우터 수신율을 측정하였다 데이터 패킷의 특성은 시뮬레이션에서 지정하였다. 시뮬레이션 도구는 멀티캐스트 환경을 잘 반영하는 ns-2 (Network Simulator-II)를 사용하였고 시뮬레이션의 공정성을 기하기 위해 랜덤 값을 이용하여 각 방식으로 100번을 반복 수행한 후 평균을 결과 값으로 채택하였다.

그림5는 송신자로부터 멀티캐스트 그룹 멤버에 이르는 패킷을 생성, 전송하고 각각의 라우팅 방식에 대하여멀티캐스트 라우터에서의 패킷에 따른 버퍼 관리를 측정하였다.

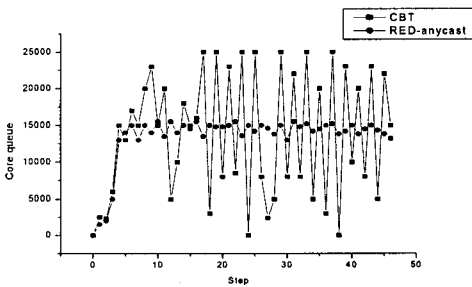


그림5 멀티캐스트 라우터에서의 버퍼 관리

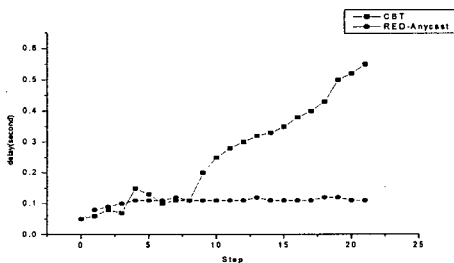


그림6 패킷크기가 512 Byte 일 때 라우터 평균 지연 시간의 측정

그림6는 토폴로지에 대하여 평균 지연 시간의 측정 결과이다. Anycast 라우팅과 비교하여 다소 우수함을 알 수 있다. 그러나 시간이 지나면서 CBT라우팅의 지연시간이 원만하게 증가하는 반면, Anycast라우팅의 경우 그룹 수의 증가에도 큰 변화를 나타내지 않았다. CBT 라우팅에 대하여 Anycast 라우팅의 지연특성은 각각 14.4% 개선으로 나타났으며 다양한 멀티미디어 패킷의 크기 변화에 따른 Anycast의 지연 시간은 평균 8.7% 감소하였다.

4결론

본 논문에서는, 비교적 낮은 대역폭에서 안정적인 CBT 공유 트리 라우팅 방식으로부터, 트래픽 부하의 증가에 따라 높은 대역폭에서도 트래픽의 분산에 적합한 Anycast 라우팅 방식로의 전환을 위한 방안을 제안하였다. 이를 위해 멀티캐스트 라우팅 방식의 고찰을 선행하고 CBT 라우팅 방식의 단점을 고찰하였으며, RED 큐잉 모델을 적용하여 이론적으로 분석하고 큐잉 모델로 일반화하여 CBT 코어 라우터의 큐잉지연을 측정하였다.

참고문헌

1. A. Ballardie, "Core Based Trees (CBT) Multicast Routing Architecture", RFC2201, 1997.
2. A. Ballardie, "Core Based Trees (CBT Version 2) Multicast Routing Protocol Specification RFC2189, 1997.
3. J.Moy, "Multicast Extensions to OSPF", IETF RFC 1584, 1994.
4. K. Ettikan, "An Analysis Of Anycast Architecture And Transport Layer Problems", Asia Pacific Regional Internet Conference on Operational Technologies, Kuala Lumpur, Mal-aysia, Feb.,-March, 2001.
5. J. Lin and S. Paul, "RMTP: A Reliable Multicast Transport Protocol," IEEE INFOCOM '96, San Francisco, CA, March 1996.R. Yavatkar, J. Griffioen, and M. Sudan, "Reliable Dissemination
6. W. Yoon, D. Lee, H.Youn, S. Lee, S. Koh, "A Combined Group/Tree Approach for Many-to-Many Reliable Multicast," IEEE INFOCOM'02, June 2002
7. Christiansen,M.,Jeffay,K.,Ott,D.,and Smith F.D., "Tuning RED for Web Traffic,"IEEE/ACM Transactions,June,2001
8. Floyd, S., "Ns Simulator Testrs for Random Early Detection(RED) Gatewqys," Technical Report, October, 1996.
9. "The Network Simulator: ns-2," <http://www.isi.edu/nsnam/ns/>