DOI: 10.3745/KIPSTC.2011.18C.1.051

라이브 멀티미디어 스트리밍 서비스를 위한 P4P 프레임워크 기반의 P2P 오버레이 시스템

변 해 선[†] · 이 미 정[††]

요 약

본 논문에서는 라이브 멀티미디어 스트리밍의 서비스 품질을 지원할 수 있는 P4P(Proactive network Provider Participation for P2P) 기반의 P2P(Peer-to-Peer) 시스템을 제안한다. 라이브 멀티미디어의 엄격한 지연 요구를 지원하기 위해, 제안하는 방안에서는 P4P의 네트워크 제공자 측 서버가 네트워크 자원 활용 최적화를 위한 정보와 더불어 지연 및 혼잡 링크에 관한 네트워크 상태 정보를 수집하여 이를 명시적으로 P2P 시스템에 제시하도록 하고, 이에 기반하여 P2P 시스템 측 서버가 피어링 제의를 수행하도록 하였다. 또한 모든 피어들이 소스 피어의 재생 지점을 기준으로 일정 범위 내에서 재생을 시작하도록 하는 라이브 멀티미디어 스트리밍 재생 동기화 방안을 제안한다. 시뮬레이션을 통해 성능을 평가한 결과, 제안 방안은 P4P의 기본 목적인 네트워크 자원 활용 문제를 효과적으로 다루면서 라이브 멀티미디어 스트리밍 서비스의 지속성을 향상시키고, 재생 시작 지연 및 제어 오버헤드를 줄임을 확인하였다. 또한 시스템 전체 피어들 간 재생 지점의 편차를 줄임으로써 실시간성을 개선함을 볼 수 있었다.

키워드 : 라이브 멀티미디어 스트리밍, P2P 오버레이 네트워크, P4P 프레임워크, 재생 동기화

A P2P Overlay System based on P4P-framework for Live Multimedia Streaming Services

Haesun Byun[†] · Meejeong Lee[††]

ABSTRACT

In this paper, we propose a P4P based P2P system for live multimedia streaming services. In order to satisfy the strict requirement of delay in live multimedia streaming, in the proposed scheme, the P4P server of network provider provides the network status information related to delay and congestion links to P2P system in addition to the information to optimize the network resource utilization. The P2P system server, then, makes the peering suggestion based on the information from the network server. Also, we propose a playback synchronization mechanism that enable each peer to start the playback within the limited variation from the playback positions of source peer. Through the simulation results, it is shown that the proposed scheme not only deals with the original objective of the P4P framework, i.e., effective network utilization, but also the live multimedia streaming requirements. It enhances the playback continuity, and reduces the playback start-up latency and the control overhead. In addition, the proposed scheme reduces the variation in playback positions of the peers.

Keywords : Live Multimedia Streaming, P2P Overlay Network, P4P framework, Playback Synchronization

1. 서 론

최근 다수의 인터넷 트래픽에 대한 조사 연구에서 P2P 트래픽이 네트워크 트래픽의 50~70%를 차지하고 있음이 보고된바 있고, 특히 동영상 콘텐츠의 트래픽 점유율이 90%에 이르게 될 전망임이 보고된 바 있다[1][2]. 지금까지 제안된 대부분의 P2P 시스템들은 네트워크 자원 사용에 제약이 없다는 가정 하에 P2P 어플리케이션 측면에서의 성능 향상을 고려하여 디자인되었기 때문에, 네트워크 서비스 제공자 입장에서는 효율적이고 공정한 네트워크 자원 활용이 어렵다는 문제를 가지고 있다. 또한 이와 같은 문제를 해결하기 위해 네트워크

※ 이 논문은 2010년도 정부(교육과학기술부)의 재원으로 한국연구재단의 지원을 받아 수행된 연구임(No. 2010-0000269).
† 준 회 원 : 이화여자대학교 대학원 컴퓨터학과 박사과정
†† 정 회 원 : 이화여자대학교 공과대학 컴퓨터학과 교수
논문접수 : 2010년 10월 28일
수 정 일 : 1차 2010년 12월 13일
심사완료 : 2010년 12월 21일

서비스 제공자가 P2P 트래픽에 제재를 가하는 메커니즘을 도입하게 된다면 P2P 서비스 품질에 있어서도 결정적인 문제를 야기할 수 있다. 이에 DCIA(Distributed Computing Industry Association)의 P4P(Proactive network Provider Participation for P2P) 워킹그룹에서는 네트워크 서비스 제공자와 P2P 어플리케이션 간 협력 방안으로 P4P 프레임워크를 제안하였다[3]. P4P에서는 네트워크 서비스 제공자 측에서 동작하는 서버가 P2P 시스템 측의 P2P 어플리케이션 서버(트래커 기반 구조의 경우) 또는 피어(트래커-리스 구조의 경우)에게 네트워크의 상태, 네트워크 정책(policy), 네트워크 능력(capability) 등의 네트워크 정보를 제공하고, P2P 어플리케이션 서버 또는 피어는 이와 같은 네트워크 정보와 어플리케이션 요구사항을 반영하여 P2P 트래픽을 제어한다. P4P는 P2P 어플리케이션의 성능을 향상(또는 유지)시키면서 네트워크 자원을 최적으로 활용하는데 목적을 두고 있다.

기존의 P4P 연구에서는 네트워크상에 병목 영역이 발생되지 않도록 하고, 네트워크 내 유통되는 P2P 트래픽 양을 최소화하기 위해 최대 링크 활용율 (MLU: Maximum Link Utilization) 혹은 링크의 트래픽 양과 피어링이 이루어지는 PoP(Point of Presence)간 홉수의 곱 (BDP: Bandwidth-Distance Product)을 최소화하도록 유도하는 비용을 P2P 시스템에 제공하는 방안을 제시하였다. 그리고, 이와 같은 피어링 비용이 주어졌을 때 각 PoP 별 업로드와 다운로드 용량(capacity) 제약 하에 서로 다른 AS(Autonomous System) 간의 피어링 비율을 줄일 수 있도록 하는 방안을 제안함으로써 네트워크 자원 활용 효율을 높일 수 있도록 하였다.

MLU 혹은 BDP의 최소화를 달성하여 네트워크에 병목 영역이 발생되는 것을 피하고 P2P 트래픽이 점유하는 네트워크 자원을 최소화하도록 피어링을 함으로써 P2P 자체도 데이터 전달 지연을 줄이고 네트워크 병목으로 인한 서비스 중단 등의 이상 현상을 피함으로써 서비스 성능을 향상시킬 수 있지만 라이브 멀티미디어 스트리밍의 경우는 만족스러운 서비스를 위해 이 보다 좀 더 직접적으로 지연에 관한 제약을 피어링에 반영할 필요가 있다. 가령 MLU를 만족하는 피어링이 경우에 따라서는 라이브 멀티미디어 스트리밍의 지연(10초 이하의 종단간 지연) 혹은 손실율(1% 미만의 손실율) 요구를 만족시키지는 못할 수 있기 때문이다. 이에 본 논문에서는 서비스 품질에 미치는 지연과 손실율의 영향이 일반적인 데이터보다 훨씬 민감한 라이브 멀티미디어 스트리밍을 위해, P4P가 링크 활용율과 PoP 간의 지연을 명시적으로 제공하도록 하고 이를 P2P 시스템의 피어링 결정에 반영하도록 함으로써 P4P의 기본 목적인 네트워크 자원 활용 향상과 더불어 라이브 멀티미디어 스트리밍 P2P 시스템을 효과적으로 지원할 수 있도록 하는 방안을 제안하였다. 또한, 동일 PoP 내에서 피어링이 이루어지는 경우, 각 피어 별 업로드 다운로드 용량 제약을 반영함으로써 지연 측면에서 가장 유리한 PoP 내 피어링이 효율적으로 이루어질 수 있도록 하였으며, 지속적인 스트리밍 서비스를 위해 피어의 스트림 가용성을 반영함으로써 스트림을 많이 확보하고 있는 피어를 우선적으로 피어링에 활용하도록 하였다.

스트리밍의 지속성, 시스템의 안정성 등과 함께 라이브 멀티미디어 스트리밍의 서비스 품질에 영향을 미치는 요소로 피어간 재생 동기화가 고려되어야 한다. 재생 동기화는 라이브 스트리밍의 실시간성 보장에 관련된 지표가운데 하나로서 이를 위해 기존의 라이브 멀티미디어 스트리밍 P2P 시스템에서는 피어가 파트너들과 스트림 가용 정보인 버퍼맵을 주기적으로 교환하고 메시 기반으로 스트림을 전달하는 데이타-드리븐 구조를 취하고 있다[4][5][6]. 데이타-드리븐 구조는 자신의 상대 피어(부모와 파트너 피어)들의 버퍼맵에 있는 서브스트림을 기준으로 일정 범위 이전 지점을 스트림 요청 시작 지점으로 결정한다. 데이타-드리븐 방안의 이러한 재생 동기화 방법은 언제나 상대 피어가 확보하고 있는 스트림을 기준으로 최초 요청 지점을 결정하기 때문에 초기 버퍼링을 위해 필요한 시간을 단축시킬 수 있다는 장점을 가지나, 소스 피어의 재생 지점과 상관없이 상대 피어의 버퍼맵을 기준으로 최초 요청 지점을 결정하므로 P2P 오버레이 시스템이 큰 규모일 경우 참여하고 있는 피어들 간 재생 지점의 편차가 커질 수 있다. 스포츠 생중계와 같은 동시성을 요구하는 서비스는 서비스를 이용하는 피어들 간 서로 동일하지 않는 화면을 보면서 사용자간 인터랙션에 문제가 발생할 수 있기 때문에 재생 지점 편차 증가는 서비스의 실시간성을 저해하는 요건이다. 이와 같은 문제를 해결하기 위하여 본 논문에서는 스트림을 주고받는 인접 피어들간의 상대적인 재생 동기화가 아닌 소스 피어와 피어들 간 재생 지점의 차를 일정한 범위내로 제어할 수 있는 재생 동기화 메커니즘을 제안하였다.

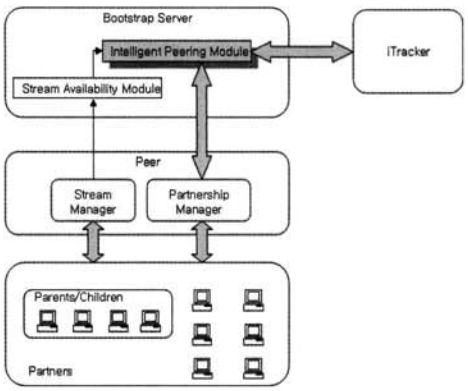
시뮬레이션을 통해, 제안하는 방안이 인트라-AS뿐만 아니라, 인터-AS간 트래픽을 감소시키고 각 링크별 트래픽 부하를 더 균등하게 유지하는 등 P4P의 기본 목적을 달성하면서도 기존의 데이타-드리븐 방안보다 라이브 멀티미디어 스트리밍 시 요구사항인 패킷 손실 측면에서 요구사항을 만족시키는 피어의 비율이 더 높음을 볼 수 있었다. 또한 소스 피어의 재생 지점을 기준으로 하는 재생 동기화 메커니즘을 통해 피어들 간 재생 지점의 편차를 줄임으로써 버퍼맵 기반으로 최초 재생 지점을 설정하는 기존의 데이터-드리븐 방안보다 라이브 멀티미디어 스트리밍 서비스의 실시간성을 향상시킴을 볼 수 있었다.

본 논문의 구성은 다음과 같다. 서론에 이어, 제 2장에서는 제안하는 P4P-프레임워크 기반의 P2P 오버레이 시스템에 대해 더 자세히 설명하고, 제 3장에서는 이에 대한 성능 평가 결과를 설명하며, 마지막으로 4장에서는 결론을 맺는다.

2. 제안하는 P4P-프레임워크 기반의 P2P 오버레이 시스템

이 장에서는 제안하는 P4P-프레임워크 기반의 P2P 오버레이 구조와 시스템 컴포넌트를 먼저 설명하고, 제안하는 방안을 구성하는 주요 메커니즘인 피어링 제의 메커니즘, 피어들 간 재생 지점의 편차를 줄이기 위한 피어들 간 재생 동기화 메커니즘, 부모 피어 갱신 메커니즘에 대해 차례로 기술한다.

2.1 P4P-프레임워크 기반의 P2P 오버레이 구조

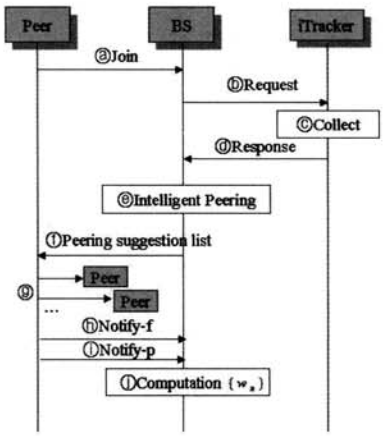


(그림 1) P4P-프레임워크 기반의 P2P 시스템 컴포넌트

(그림 1)은 제안하는 P4P-프레임워크 기반의 P2P 시스템의 컴포넌트를 보이고 있다. 제안하는 시스템은 피어, 부트스트랩 서버, P4P에서 네트워크 정보를 제공하는 ISP 측의 서버인 iTracker로 구성되어 있다. 각 피어는 P2P 오버레이 네트워크의 멤버이며, 파트너 피어 및 부모, 자식 피어 정보를 가지고 있고, 이 정보는 파트너십 매니저에 의해 관리된다. 각 피어의 스트림 매니저는 부모 피어(들)에게 스트림을 요청하며, 자식 피어(들)에게 스트림을 전달한다. 또한 버퍼를 주기적으로 검사하며 재생 시점이 임박하였는데 아직 받지 못한 세그먼트가 있다면 파트너 피어에게 이를 요청한다.

부트스트랩 서버는 P2P 시스템의 진입(entry) 노드로써 인텔리전트 피어링 모듈과 스트림 가용성 모듈을 가지고 있다. 인텔리전트 피어링 모듈은 현재 시스템에 들어와 있는 피어들 중 새롭게 조인 요청한 피어와 통신하기에 가장 적합한 피어들의 리스트를 선택하는 역할을 수행하는데, iTracker로부터 받은 네트워크 정보, 피어로부터 받은 업로드/다운로드 링크의 용량 및 스트림 가용성 모듈에 의해서 계산된 스트림 가용 정보를 이용하여 피어들을 선택한다. 스트림 가용성 모듈은 조인한 피어로부터 받은 '스트림 가용 메시지'와 피어가 시스템에 들어와 있는 시간을 파악하여 각 피어들이 스트림을 어느 정도 안정적으로 제공할 수 있는지를 주기적으로 계산한다. iTracker는 ISP의 네트워크 상태를 모니터링하는 노드로써 부트스트랩 서버에게 네트워크 상태 정보를 전달해준다.

제안하는 P2P 시스템은 CoolStreaming[4]에서와 같이 멀티플 서브스트림의 개념을 사용한다. 이 개념에서 피어는 자식 피어에게 하나 이상의 서브스트림을 전달해주는 부모 피어일 수 있으며, 일부 서브스트림에 있어서 부모 피어인 반면, 나머지 서브스트림에 있어서 파트너 피어 일 수 있다. 자식 피어는 각 부모 피어로부터 해당 서브스트림을 풀(pull)하며, 부모 피어는 자식 피어에 의해서 요청된 서브스트림의 모든 세그먼트를 푸시(push)한다. CoolStreaming과 다르게, 제안하는 방안에서는 한 피어가 해당 부모 피어로부터 서브스트림의 일부 세그먼트를 전달받지 못했다면 그 피어는 파트너 피어들 중 한 피어에게 그 세그먼트를 요청한다. 따라서 제안하는 시스템은 각각의 서브스트림이 지정된 부모 피어를 통해 푸시 메커니즘으로 전달되는, 각 서브스트림 별 트리인 멀티플 트리 오버레이와 임의의 파트너 피어를 통해 긴급 세그먼트를 요청하여 받는 보조의 메시 오버레이를 혼합한 하이브리드 구조를 띠고 있다.



(그림 2) P4P-프레임워크 기반의 P2P 오버레이 구조를 구성하기 위한 절차

(그림 2)는 제안하는 방안에서 P2P 오버레이 구조를 구성하기 위한 절차를 보인 것이다. 피어는 P2P 오버레이 시스템에 조인하기 위해 부트스트랩 서버에게 조인 요청을 한다ⓐ. 이때 피어는 부트스트랩 서버에게 IP 주소, PID(PoP의 ID), AS 번호, 업로드/다운로드 링크 용량 등을 보낸다. PID 및 AS 번호는 사용자(client)가 프로비저닝 시스템이나 iTracker로부터 획득한다[3]. 부트스트랩 서버는 조인한 피어가 속한 각 AS의 iTracker에게 네트워크 상태 정보를 주기적으로 요청하며ⓑ, iTracker는 자신이 관리하는 AS의 PID-i로부터 PID-j로 가는 경로 상에 있는 모든 링크 e에 대하여 각 링크의 라우팅 비용, 지연, 링크 활용률 상태를 주기적으로 수집하고 계산하여ⓒ, 그 정보를 부트스트랩 서버에게 전달한다ⓓ(iTracker가 네트워크 상태 정보를 계산하는 자세한 방법은 2.2절에 설명함).

부트스트랩 서버는 새롭게 조인한 피어를 위해 인텔리전트 피어링 제의 메커니즘을 수행하는데ⓔ, 이때 iTracker로부터 받은 정보와 스트림 가용성 가중치(stream availability weight)(부트스트랩 서버가 스트림 가용성 가중치를 계산하는 방법은 2.2.절에서 설명함), 피어의 업로드/다운로드 링크

용량 등을 고려하여 피어링 제의 리스트를 만들고(자세한 방안은 2.2절에서 설명함), 이를 새롭게 조인한 피어에게 전달한다ⓕ. 조인 요청한 피어는 부트스트랩 서버로부터 받은 피어링 제의 리스트의 부모 피어 및 파트너 피어와 커넥션을 설립하고 피어들 간 재생 동기화 메커니즘을 통해 스트림의 최초 요청 지점을 결정하여, 스트림을 요청한다ⓖ. 새롭게 조인한 피어는 부모 피어로부터 처음으로 세그먼트를 받으면 부트스트랩 서버에게 Notify-f 메시지를 보내 이를 통보한다ⓗ. 또한 새롭게 조인한 피어는 재생 동기화 메커니즘에서 결정한 최초 재생 예측 지점까지 버퍼링이 이루어지면 Notify-p 메시지를 부트스트랩 서버에게 보낸다ⓘ. 이 두 메시지는 부트스트랩 서버가 피어의 스트림 가용성 가중치를 계산하기 위해 사용된다. 부트스트랩 서버는 이들 메시지를 받으면 해당 피어의 스트림 가용성 가중치를 계산한다ⓙ. 스트림 가용성 가중치는 조인한 피어가 서브스트림을 획득한 후 보내는 스트림 가용 메시지(Notify-f, Notify-p)와 그 이후 시스템에 머문 시간을 가지고 주기적으로 계산된다.

2.2 피어링 제의 메커니즘

라이브 멀티미디어 스트리밍 서비스는 일반적인 데이터 서비스보다 서비스 품질에 미치는 지연과 손실율의 영향이 훨씬 민감하다. 라이브 멀티미디어 스트리밍의 만족스러운 서비스를 위해 [3]에서 제시한 피어링 비용보다 좀 더 직접적으로 지연에 관한 제약을 피어링에 반영할 필요가 있다. 이에 제안하는 피어링 제의 메커니즘에서는 iTracker가 링크 활용율과 PoP 간의 지연을 명시적으로 부트스트랩 서버에게 제공하도록 하고 이를 P2P 시스템의 피어링 결정에 반영하도록 함으로써 P4P의 기본 목적인 네트워크 자원 활용 향상과 더불어 라이브 멀티미디어 스트리밍 P2P 시스템을 효과적으로 지원할 수 있도록 하였다. 또한, 동일 PoP 내에서 피어링이 이루어지는 경우, 각 피어 별 업로드 다운로드 용량 제약을 반영함으로써 지연 측면에서 가장 유리한 PoP 내 피어링이 효율적으로 이루어질 수 있도록 하였으며, 지속적인 스트림 제공을 위해 피어의 스트림 가용성 가중치를 피어링에 반영함으로써 스트림을 많이 확보하고 있는 피어를 우선적으로 피어링에 적용하도록 하였다.

2.2.1 iTracker에서의 동작(그림 2-ⓒ)

부트스트랩 서버는 주기적으로 iTracker에게 네트워크 상태 정보를 요청한다. iTracker는 자신이 관리하는 AS의 라우팅 비용(r_{ij}), 지연(d_{ij}), 링크 활용률 상태(u_{ij})를 계산한다.

라우팅 비용은 다음 식(1)과 같이 계산된다. PID-i로부터 PID-j로 가는 경로상의 링크 e의 라우팅 비용을 r_e라 할 때 PID-i로부터 PID-j로 가는 경로 상의 모든 링크의 라우팅 비용 r_e의 합이다.

$$r_{ij} = \sum_{e \in path(i,j)} r_e \tag{1}$$

지연은 식(2)와 같이 계산된다. PID-i로부터 PID-j로 가는 경로상의 링크 e의 지연을 d_e라 할 때 PID-i로부터 PID-j로 가는 경로상의 모든 링크의 지연 d_e의 합이다.

$$d_{ij} = \sum_{e \in path(i,j)} d_e \tag{2}$$

PID-i로부터 PID-j로 가는 경로상에 링크 e의 링크 활용률 l_e는 b_e/c_e이다. 여기서 c_e는 링크 e의 대역폭이며 b_e는 링크 e를 지나는 트래픽의 양이다. l_e가 링크 활용률의 제한치인 LT_e보다 크면 그 링크를 지나는 트래픽의 양을 증가시키지 않기 위하여 그 링크를 혼잡 상태라 간주하고, 링크 활용 상태 u_e를 0으로 한다. 만약 l_e가 링크 활용률의 제한치인 LT_e보다 크지 않으면 u_e를 1로 한다. 각 링크의 u_e를 이용하여, PID-i로부터 PID-j로 가는 경로의 링크 활용률 상태는 식(3)과 같이 계산한다.

$$u_{ij} = \prod_{e \in path(i,j)} u_e \tag{3}$$

2.2.2 부트스트랩 서버에서 스트림 가용성을 계산하는 방법 (그림 2-ⓙ)

스트림 가용성이란 피어가 안정적으로 멀티미디어 스트림을 제공할 수 있는 정도를 의미한다. 본 방안에서는 스트림의 가용성을 세단계로 구분하였다. 첫 번째 단계는 피어가 조인한 이후에 부모 피어로부터 첫 번째 세그먼트를 받는 단계이다. 두 번째 단계는 피어가 재생 동기화 메커니즘에서 결정한 최초 재생 예측 지점까지 버퍼링하기까지의 단계이다. 세 번째 단계는 이 이후의 단계이며, 부트스트랩 서버는 다음과 같이 식(4)에 의해 스트림 가용성을 계산한다.

$$p_n = \frac{1}{t_{init}^n} \cdot \alpha + \frac{1}{t_c^n} \cdot \beta + t_s^n \cdot \gamma \tag{4}$$

여기서 t_{init}^n은 피어 n이 조인한 후에 부모 피어로부터 첫 번째 세그먼트를 받을 때까지 경과한 시간이고, t_c^n은 피어 n이 첫 번째 세그먼트를 받은 후에 재생 동기화 메커니즘에서 결정한 최초 재생 예측 지점까지 버퍼링하기까지 경과한 시간이다. t_s^n은 피어 n이 최초 재생 예측 지점까지 버퍼링 이후 시스템에 머무른 시간이다. α, β, γ는 각 단계별 가중치를 의미하는데, 각 단계별로 소요되는 시간의 차가 클 수 있기 때문에 각 가중치를 다르게 적용한다. 식(4)을 통해, 부모 피어로부터 짧은 시간 내에 스트림을 확보한 피어일수록, 또 시스템에 머무른 시간이 길수록 높은 스트림 가용성을 얻게 되어 부모 피어의 안정성과 피어의 지속성이 고려된 스트림 가용성이 계산된다. 부트스트랩 서버는 조인한 피어들에 대해 주기적으로 스트림 가용성 가중치(w_n)를 수식(5)와 같이 계산한다.

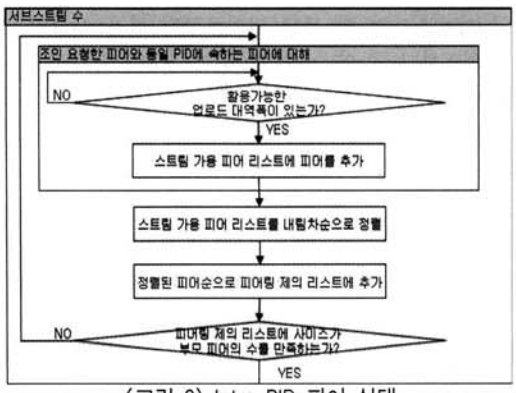
$$w_n = \frac{p_n}{\sum_{a \in A} p_a} \quad (5)$$

여기서 A는 동일 PoP에 위치해 있는 액티브 피어의 집합이다. 즉, 피어 n의 스트림 가용성 가중치는 동일 PoP에 위치해 있는 액티브 피어들의 스트림 가용성의 합에 대한 피어 n의 스트림 가용성의 비중이다.

2.2.3 부트스트랩 서버가 피어링 제의 리스트를 만드는 방법(그림 2-ⓔ)

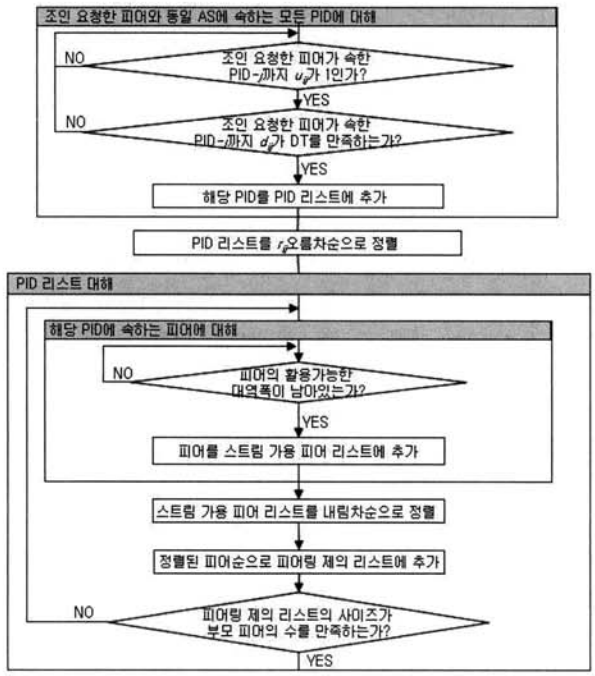
부트스트랩 서버는 스트림 가용성 가중치, 피어의 업로드/다운로드 링크 용량, iTracker로부터 받은 라우팅 비용, 지연, 링크 활용률 상태를 이용하여 조인 요청한 피어와 통신할 최적의 피어를 선택한다. 부트스트랩 서버에서의 피어 선택 방법은 다음과 같다. 제안하는 방안에서는 피어링을 위한 시스템 파라미터로 m, s, k를 사용한다. m은 파트너 피어의 수이다. s는 서브스트림의 수이고, k는 부모 피어로 선택된 피어의 수이다. 동일 피어가 하나 이상의 서브스트림에 대한 부모 피어로 선택될 수 있으므로 $s \geq k$이다.

PID-j에 조인한 피어에 대하여 부트스트랩 서버는 각 서브스트림 별 부모 피어를 선택한다. 즉, s번의 부모 피어 선택을 수행하는데 먼저, 조인 요청한 피어와 동일 PoP, 그리고 동일 AS내의 타 PoP, 마지막으로 타 AS 순으로 선택해 나간다. 먼저 조인 요청한 피어와 동일 PoP에 속하는 피어들 가운데서는 활용 가능한 업로드 대역폭을 가진 피어들중 스트림 가용성 가중치가 높은 순서대로 라운드 로빈으로 선택한다(그림 3). 이를 통해 s개의 피어가 모두 선택되지 못한 경우 동일 AS내에서 inter-PID 피어 선택을 수행한다. Inter-PID 선택에서는 u_{ij}가 1인 PoP에 대해 d_{ij}가 지연 제한치인 DT(delay threshold)보다 적은 PoP들을 r_{ij} 오름차순으로 정렬하고, 라우팅 비용이 낮은 PoP 순으로 피어를 찾아나간다(그림 4). 이와 같은 동일 AS내 inter-PID 피어 선택에서도 s에 이르는 피어를 선택하지 못했다면 inter-AS 피어 선택을 수행한다. Inter-AS 피어 선택은 조인한 피어가 속해 있는 ISP가 선호하는 AS를 우선적으로 선택하도록 한다.



(그림 3) Intra-PID 피어 선택

s개의 서브스트림별 부모 피어 선택에 의해 k개의 부모 피어가 선택되면 추가적으로 $(m-k)$개의 파트너 피어를 선택한다. 파트너 피어 선택은 부모 피어 선택과 같은 방식으로 이루어지되 단, 피어의 가용 대역폭은 고려하지 않는다. 그 이유는 파트너 피어는 부모 피어가 보낸 서브스트림에서 누락된 세그먼트가 발생했을 때 이를 보완하는 피어이기 때문에 지속적인 대역폭 요구는 없기 때문이다.



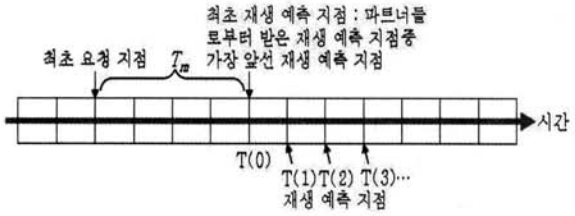
(그림 4) Inter-PID 피어 선택

2.3 피어들 간 재생 동기화 메커니즘

상대 피어(부모 피어와 파트너 피어)의 버퍼에 있는 최대 세그먼트 번호로부터 T_p개(T_p는 시스템 파라미터) 이전의 세그먼트를 스트림의 최초 요청 지점으로 결정하는 데이터-드리븐 방안들의 경우, 부모 피어 p가 자식 피어 q에게 전송할 수 있는 업로드 대역폭이 p가 한 서브스트림을 전송하는데 소요되는 평균 전송률보다 훨씬 크다면, q는 p가 버퍼링한 지점까지 짧은 시간 안에 버퍼링 할 수 있기 때문에 소스 피어와의 재생 지점의 차이가 크지 않다. 그러나 p의 업로드 대역폭이 서브스트림을 전송하기 위해 요구된 평균 전송률과 거의 비슷하거나 작아 q의 버퍼링 속도가 느린 경우, q가 p의 최대 버퍼링 지점까지 버퍼링하지 못한 상태에서 새롭게 조인한 다른 피어 n에 대해서 부모 피어 또는 파트너 피어로 선택 된다면, 새롭게 조인한 피어 n은 p의 버퍼링 지점보다 훨씬 뒤쳐지는 지점의 버퍼맵을 q로 부터 전달받는다. 이러한 상황은 피어들이 P2P 시스템에 한꺼번에 접속할 때 빈번하게 발생하게 되는데, 이러한 경우, 조인 요청한 피어들은 소스 피어보다 훨씬 뒤쳐지는 지점의 버퍼맵을 부모 피어 또는 파트너 피어로 부터 전달받게 되어 소스 피어의 재생 지점과 큰 차를 갖고 재생하게 된다. 재생 도

중, 버퍼링이 더 많이 진행된 파트너 피어를 발견하여 그 파트너 피어를 부모 피어로 갱신하고 스트림을 건너 뛰어 재생위치를 조정하게 되면 재생 지점의 차를 감소시킬 수 있으나, 새롭게 조인한 피어 n의 부모 피어와 모든 파트너 피어가 소스 피어의 재생 지점보다 뒤쳐지는 지점의 버퍼맵 정보를 지속적으로 전달하는 경우에는 소스 피어와 재생 지점의 차가 큰 상태로 계속 재생하게 된다.

라이브 멀티미디어 스트리밍 서비스에서 동일 시간대에 다수의 사용자가 접속하여 서비스를 요청할 수 있는데 이러한 상황에서도 피어들 간 재생 지점이 적정하게 유지될 수 있도록 하는 것은 서비스의 실시간성을 위해 매우 중요하다. 이를 위해 모든 피어들 간 소스 피어의 재생 지점을 공유하고, 이에 따라 재생 시작 지점을 결정할 수 있는 메커니즘이 필요하다. 이에 제안하는 방안에서는 각 피어가 소스 피어의 재생 지점을 예측하는 방법을 이용한다. 즉, 각 피어는 조인 시점에 파트너 피어들로부터 그들이 예측하고 있는 소스 피어 재생 지점을 전달받고, 이를 기반으로 시스템에 참여하고 있는 동안 지속적으로 소스 피어의 재생 지점을 예측하면서 자신의 파트너 피어에게 예측한 재생 지점을 알려준다. 제안하는 방안에서 각 피어는 소스 피어의 재생 지점을 스트림에서 발생하는 초당 프레임의 수로 예측하며, 세그먼트 번호로 재생 예측 지점을 명시한다. 즉, 소스의 재생 시작 시점으로부터 t초가 경과했고, 스트림의 프레임 율이 f/sec이며, 프레임당 평균 세그먼트 개수가 l이라면, 그 시점의 소스 재생 지점에 해당하는 세그먼트 번호는 $(t*f*l)$이 된다. 새로 조인하는 피어는 자신의 파트너 피어들로부터 재생 예측 지점을 전달 받게 되는데, 이때 전달받은 재생 예측 지점이 소스 피어와 각 피어들 간 종단간 지연으로 인해 약간의 차이가 있을 수 있다. 제안하는 방안에서는 (그림 5)에서와 같이, 수신한 재생 예측 지점 중 가장 앞선 재생 예측 지점을 새로 조인하는 피어의 최초 재생 예측 지점으로 선택한다. 그리고 최초 재생 예측 지점으로부터 T_m 이전 지점을 스트림의 최초 요청 지점으로 결정하고, 그 지점부터 세그먼트를 보내줄 것을 부모 피어에게 요청한다.



(그림 5) 최초 재생 예측 지점, 최초 요청 지점, 진행된 재생 예측 지점의 관계

2.4 부모 피어 갱신 메커니즘

P2P 시스템에서는 피어가 갑작스럽게 시스템을 탈퇴하거나 불충분한 업로드 대역폭으로 지속적인 서비스를 해주지 못하는 경우가 발생할 수 있다. 따라서 이를 사전에 발견하고 새로운 상대 피어를 선택하는 것이 매우 중요하다. 이를 위해서는 상대 피어의 탈퇴나 불충분한 업로드 대역폭으로 서비스가 지속되지 못하는 것을 전송이 이루어지지 않고 있는 세그먼트의 재생 시점 도래 이전에 감지하는 방안과 새로운 상대 피어를 선택하는 방안이 필요하다.

버퍼맵 기반의 데이터-드리븐 방안은 부모 피어와 파트너 피어들로부터 수신한 버퍼맵을 모니터링 하여 사전에 서브스트림의 지연을 발견하고 새로운 부모 피어를 선택한다. 이 방법은 부모 피어와 자신과의 서브스트림 지연의 편차, 부모 피어와 파트너 피어간의 서브스트림 지연의 편차를 활용하며, 특정 서브스트림의 지연 및 부모 피어에서의 서브스트림 지연을 사전에 발견할 수 있다는 장점을 가지고 있으나, 부모 피어 및 파트너 들 간 버퍼맵 교환 오버헤드가 매우 크다[7][8].

제안하는 방안에서는 버퍼맵을 사용하지 않는 대신 현재 수신한 서브스트림의 상태를 모니터링하여 서브스트림을 제대로 제공해주지 못하는 부모 피어를 발견한다. 피어는 부모 피어에게 스트림의 최초 요청 지점의 세그먼트를 요청한 후 버퍼링되어야 할 최소 세그먼트의 양인 T_b가 버퍼링되면 그때부터 버퍼 검사를 시작하고, 그 이후로는 매 버퍼 검사 인터벌 T_{init}마다 버퍼 검사를 수행한다. x번째 버퍼 검사에서 검사대상이 되는 버퍼의 영역은 B^x_{srt} 로부터 B^x_{end} 이며 이들 값은 다음과 같이 결정된다.

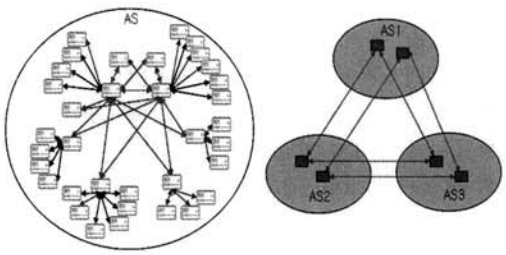
$$B^x_{srt} = \text{최초 요청 세그먼트 번호}, \text{if } x = 1$$
$$B^x_{srt} = B^{x-1}_{end} + 1 \qquad otherwise,$$
$$B^x_{end} = B^x_{srt} + T_{init} \times f \times l.$$

버퍼 검사 영역 내에서 아직 받지 못한 세그먼트가 있다면 그 세그먼트를 임의의 파트너 피어에게 요청한다. 피어는 버퍼 검사를 통해 각 파트너 피어가 자신의 새로운 부모 피어로 동작 할 수 있는지 여부와 부모 피어가 지속적으로 서비스를 제공해 주고 있는지 여부를 동시에 판단한다. 파트너 피어가 자신의 새로운 부모 피어로 동작 할 수 있는지의 타당성을 판단하기 위하여 T_{init}동안 그 파트너 피어에게 요청한 세그먼트 가운데 성공적으로 받은 세그먼트의 비율 h를 서브스트림별로 계산한다. 또한, 부모 피어가 지속적으로 서비스를 제공해 주고 있는지 여부를 검사하기 위하여 T_i동안 서브스트림 별 부모 피어로부터 받은 세그먼트의 수 g를 카운트한다. 만약, g가 T_{init}동안 부모피어가 제공해주어야 할 최소의 세그먼트 기준치인 T_g를 초과하지 못하였고, 연속적으로 T_g를 초과하지 못한 횟수가 부모 피어 변경을 해야 할 기준치 T_t를 초과하였다면 해당 서브스트림에 대해 새로운 부모 피어를 선택한다.

특정 서브스트림에 대해 새로운 부모 피어의 선택이 필요한 경우, 피어는 먼저 파트너 피어들 가운데 해당 서브스트림에 대한 h가 파트너 피어를 부모 피어로 선택될 수 있는 기준치 T_h를 초과하는 피어를 해당 서브스트림을 위한 새로운 부모 피어로 선택한다. 기준을 만족하는 파트너 피어가

없다면, 피어는 부트스트랩 서버에게 이전 부모 피어의 ID를 알려주고 새로운 부모 피어를 요청한다. 부트스트랩 서버는 피어의 부모 피어 변경정보를 통보받고 유지하며 이를 추후의 부모 피어 선택에 반영한다.

3. 성능평가



(그림 6) 시뮬레이션 네트워크 토폴로지

제안하는 방안의 성능을 평가하기 위해 NS-2[9](Network Simulator)를 이용하여 시뮬레이션을 수행하였다. NS-2는 TCP, UDP, FTP, HTTP 등등의 다양한 네트워크 프로토콜들을 시뮬레이션 할 수 있도록 만들어진 네트워크 시뮬레이터 툴이며, 이 시뮬레이터에 제안 방안의 프로토콜 및 동작 모듈들을 구현하였다. 또한 제안하는 방안의 성능 비교를 위해 대표적인 데이터-드리븐 방안인 CoolStreaming을 구현하였다. (그림 6)은 시뮬레이션에 사용된 네트워크 토폴로지를 보이고 있다. 시뮬레이션 네트워크는 세 개의 AS로 구성되어 있으며, 각 AS내의 대표 라우터를 다른 AS내의 대표 라우터와 연결하였고(그림 6의 오른쪽), 각 AS내의 토폴로지는 KT-VPN 토폴로지로 구성하였다(그림 6의 왼쪽). 부트스트랩 서버는 시스템 내 하나만 존재하며, iTracker는 각 AS별로 하나씩 둔다고 가정하였다. 한 AS내에 라우터의 개수는 36개이며, 각 라우터를 PoP로 간주하고, 각 라우터에 피어가 연결된다. 피어와 라우터간의 링크 지연은 0.1~0.2ms 사이의 랜덤 지연, 라우터와 라우터 간 링크 지연은 0.5~1ms 사이의 랜덤 지연으로 설정하였다. 라우터 간 대역폭은 200Mbyte로 설정하였으며, 소스 피어, iTracker, 부트스트랩 서버의 대역폭은 100Mbyte로 설정하였다. P2P 시스템은 100Mbyte, 10Mbyte, 1Mbyte의 업로드 대역폭을 가진 피어들과 그리고 전혀 업로드를 하지 않는 피어들로 구성된다고 가정하였고, 100Mbyte 업로드 대역폭을 가진 피어 10%, 10Mbyte 업로드 대역폭을 가진 피어 20%, 1Mbyte 업로드 대역폭을 가지는 피어 20%, 업로드 대역폭을 제공하지 않는 피어 50%로 설정하였다. 이 설정은 [4]와 [10]에서 측정한 사용자의 접속 위치에 대한 타입(직접 연결, UPnP, NAT, Firewall)별 피어 분포와 피어별 대역폭 분포를 따른 것이다. 또한, [11]의 연구결과에 따라 피어들의 조인이 포아송 분포를 따른다고 가정하였다.

시뮬레이션에 사용한 멀티미디어 스트리밍 파일은 초당 30프레임, 약400Kbps 속도로 재생된다. 한 스트림 파일을 4개의 서브스트림으로 나눴으며, 제안하는 방안에서 조인 시 스트림의 최초 요청 지점을 결정하기 위해 사용되는 파라미터인 T_m은 20초로 설정하였다. 또한 15초(T_b)간 재생할 수 있는 스트림이 버퍼링 되면 버퍼 검사를 시작하며, 20초간 재생할 수 있는 스트림이 버퍼링 되면 재생을 시작하는 것으로 설정하였다. 버퍼의 크기는 60초 동안 재생될 스트림을 버퍼링할 수 있는 크기이다. CoolStreaming에 관련된 파라미터는 [4]를 따랐으며, 그 밖의 시뮬레이션에서 사용된 파라미터 설정은 <표 1>과 같다.

시뮬레이션에서는 라이브 멀티미디어 스트리밍 어플리케이션 성능, P2P 시스템 성능, 네트워크 성능 등 세 가지 측면에서 성능을 평가하였다. 어플리케이션 성능에서는 평균 스트림 지속성과 재생 시작 지연, P2P 시스템 성능측면에서는 제어 오버헤드와 소스 피어와 피어간 재생 지점의 차, 네트워크 성능에서는 백본 네트워크 링크를 경유하는 트래픽의 양과 그 링크들의 트래픽 양의 편차, 인터-AS간 트래픽 양을 측정하였다.

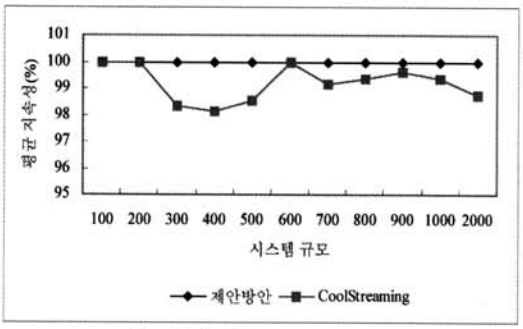
3.1 P2P 어플리케이션 성능 평가

P2P 어플리케이션의 성능을 평가하기 위해 시스템 규모(P2P 시스템에 참여하는 피어의 수)를 변경해 보면서 평균 스트림 지속성과 재생 시작 지연을 측정해 보았다. (그림 7)은 시스템 규모에 대한 평균 스트림 지속성을 보인 것이다.

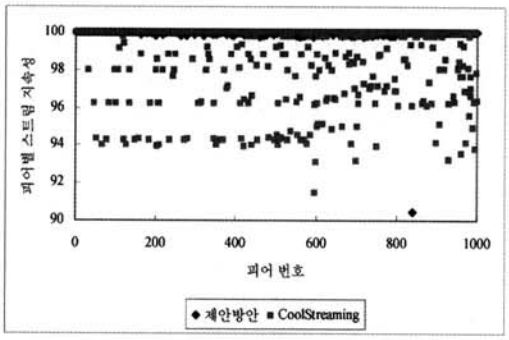
〈표 1〉 그 밖의 시뮬레이션 파라미터

네트워크 관련 파라미터	iTracker가 각 링크 활용률을 수집하는 인터벌	5초
	iTracker가 각 링크 지연을 수집하는 인터벌	5초
	AS내에 있는 PoP 간 라우팅 비용 계산	토폴로지 구성 시 한번 이루어짐. Link state 프로토콜에서의 각 PoP간 경로상에 있는 각 링크 비용의 합
	혼잡으로 간주하는 링크 활용률 제한치	링크 대역폭의 70%[12,13]
	iTracker가 부트스트랩 서버에게 네트워크 상태 정보를 전달하는 인터벌	5초
피어 관련 파라미터	버퍼 검사 인터벌 T_{init}	1초
	T_{init}동안 부모피어가 제공해주어야 할 서브스트림 별 최소의 세그먼트 기준치인 T_g	초당 해당 서브스트림 별 받아야 할 세그먼트 수의 25%
	부모 피어 변경을 해야 할 기준치인 T_t (연속적으로 T_g을 초과하지 못한 횟수에 대한 기준치)	10회
	파트너 피어를 부모 피어로 선택할 수 있는 기준치 T_h	100%

평균 스트림 지속성은 피어가 조인하여 재생을 시작한 시점부터 재생이 끝날 때까지 받아야 할 세그먼트 수 가운데 재생 데드라인을 만족시킨 세그먼트의 수에 대한 비율이다. (그림 7)에서 보는 바와 같이, 제안하는 방안의 평균 지속성은 99.9%, CoolStreaming은 98% 이상으로 두 방안 모두 높은 평균 지속성을 유지한다. 그러나 (그림 8)에서1000개의 피어에 대해 각 피어별 스트림 지속성 분포를 살펴보면 제안하는 방안은 거의 모든 피어들이 1% 미만의 전체 세그먼트 손실을 가진데 반해, CoolStreaming에서는 약 18% 정도의 피어가 1% 이상의 패킷 손실을 경험하는 것을 볼 수 있다. CoolStreaming에서는 부트스트랩 서버가 피어 정보나 네트워크 정보 없이 랜덤으로 시스템에 들어와 있는 피어들의 리스트를 알려주는 반면, 제안하는 방안에서는 iTracker로 부터 받은 네트워크 상태 정보와 함께 피어의 스트림 가용성 가중치와 피어의 업로드/다운로드 링크 용량 등을 모두 이용하여 피어링을 제의한다. 따라서 제안 방안의 피어는 최적의 피어와 통신할 가능성이 더 높고, 더 안정적으로 서비스를 받을 수 있게 된다. 결과적으로, 제안하는 방안의 피어링 메커니즘이 라이브 멀티미디어 스트리밍 서비스의 지속성을 높임을 알 수 있다.

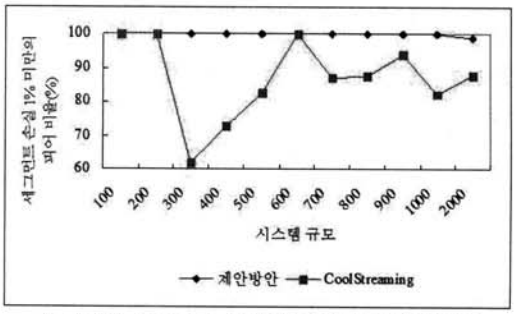


(그림 7) 평균 스트림 지속성



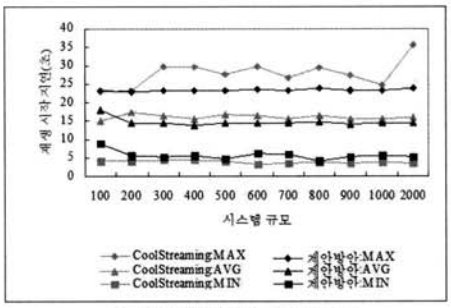
(그림 8) 1000개의 피어에 대한 각 피어별 스트림 지속성

(그림 9)는 시스템 규모를 변화시켜보면서 라이브 멀티미디어의 손실율 제약인 세그먼트 손실 1% 미만의 피어율을 보인 것이다. 두 방안 모두 평균 스트림 지속성은 높았던 반면, 세그먼트 손실 1% 미만의 피어율은 제안하는 방안에서는 거의 모든 피어들이 세그먼트 손실 1% 미만으로 스트림을 수신한 반면, CoolStreaming에서는 60% 피어들만이 세그먼트 손실 1% 미만으로 스트림을 수신한 경우도 있었다. 즉, CoolStreaming에서는 피어들간 수신율의 편차가 크다는 것을 의미한다. 피어 1000개에 대한 피어들간 스트림 수신율의 편차를 비교해 보았을 때 제안하는 방안에서는 0.30인데 반해, CoolStreaming에서는 1.55의 편차를 보였으며 60% 피어들만이 세그먼트 손실 1% 미만으로 스트림을 수신한 경우였던 피어 300개에서는 2.45의 편차를 보였다. 결과적으로 제안하는 방안이 라이브 멀티미디어 스트리밍 서비스에서 손실률 1% 미만을 만족시키는 피어의 율을 높임으로써 ITU-T G.1010 권고안 1% 미만의 패킷 손실률에 대한 요구사항을 잘 반영하고 있음을 볼 수 있었다.



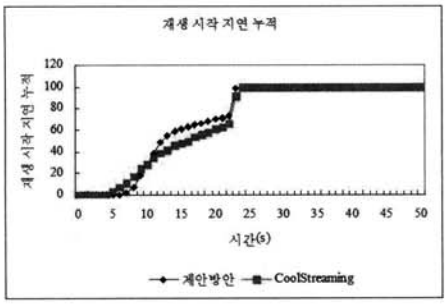
(그림 9) 세그먼트 손실 1% 미만의 피어 비율

(그림 10)은 시스템 규모를 변화시켜보면서 재생 시작 지연을 측정한 것이다. 재생 시작 지연은 피어가 조인 한 후 재생을 시작하기까지의 시간이다. (그림 10)에서 보는 바와 같이, 재생 시작 지연의 최소값은 제안하는 방안이 CoolStreaming 방안보다 좀 더 큼을 볼 수 있다. 그 이유는 CoolStreaming 방안이 제안하는 방안보다 더 이전의 지점을 스트림의 최초 요청 지점으로 요청하므로 CoolStreaming 방안의 부모 피어가 20초 연속적으로 재생할 수 있는 스트림(서브스트림 차원)을 모두 가지고 있을 가능성이 크고, 재생 시작까지를 빨리 버퍼링 할 수 있다. 반면, 재생 시작 지연의 평균값과 최대값 측면에서는 제안하는 방안이 CoolStreaming 방안보다 적음을 볼 수 있다. 이는 부모 피어가 20초 연속적으로 재생할 수 있는 스트림을 가지고 있지 않을 확률 또한 CoolStreaming 방안이 제안 방안보다 높다는 것을 의미한다. 제안 방안의 경우, 피어들간 재생 시작 지연의 편차는 9.03인데 반해, CoolStreaming에서는 12.51의 편차를 보였다. 제안하는 방안에서는 부트스트랩 서버가 피어의 스트림 가용성 가중치와 피어의 대역폭을 기반으로 피어링을 제의해주기 때문에 스트림을 지속적으로 서비스 해줄 수 있는 피어를 부모피어로 선택할 가능성이 크다. 반면, CoolStreaming에서 어떤 부모 피어는 서비스를 잘 해주는 반면 어떤 부모 피어는 서비스를 제대로 해주지 못하여 재생 시작 지연의 편차가 크다는 것을 알 수 있다.



(그림 10) 재생 시작 지연

(그림 11)은 피어 1000개에 대한 재생 시작 지연의 누적 그래프를 보인 것이다. CoolStreaming에서 약 28% 정도의 피어가 10초 이내에 재생을 시작하여 제안방안보다 재생 시작 지연이 짧은 반면, 10초 이상에 대해서는 재생을 더 빨리 시작하는 피어가 제안 방안에서 더 많음을 볼 수 있다.



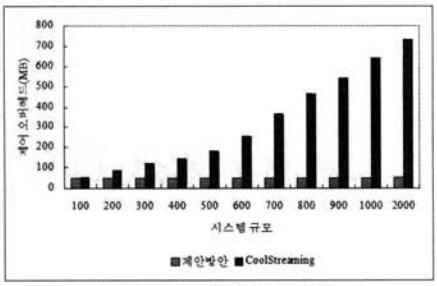
(그림 11) 재생 시작 지연 누적

3.2 P2P 시스템 성능 평가

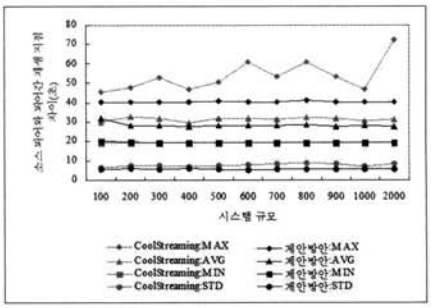
P2P 시스템 차원의 성능을 평가하기 위해 제어 메시지 오버헤드, 피어들 간 재생 지점의 편차를 측정해보았다. (그림 12)는 시스템 규모에 대한 피어에서의 제어 오버헤드를 보인 것이다. 여기서 제어 오버헤드는 시스템에 참가하는 모든 피어가 발생한 제어 메시지 양의 합을 나타낸다. 그림 12에서 보는 바와 같이, CoolStreaming 방안에서의 제어 오버헤드는 시스템 사이즈가 증가할수록 선형적으로 커지는 반면, 제안 방안은 시스템 사이즈에 큰 영향을 받지 않음을 볼 수 있다. CoolStreaming의 경우 버퍼맵을 교환하기 위한 제어 메시지 및 피어를 발견하기 위한 가십 기반의 프로토콜의 오버헤드로 인해 제어 오버헤드가 증가하며 이는 시스템 사이즈와 피어별 파트너 수 증가에 영향을 받는다[7].

(그림 13)은 시스템 규모에 대한 소스 피어와 피어간 재생 동기화 정도를 보인 것이다. (그림 13)에서와 같이, 제안방안이 CoolStreaming 방안보다 소스와 피어의 재생 지점 차에 대한 성능이 최대값, 평균값, 최소값 부분에서 더 좋은 성능을 보였는데, CoolStreaming 방안의 소스와 피어간 재생 지점의 차이가 제안 방안의 최악의 경우에도 최대값 1.7배, 평균값 1.02배, 최소값 1.44배에 달한다.

재생 지점의 차이는 스트림의 최초 요청 지점과 재생 시작 지연의 영향을 받는다. 제안방안은 소스 피어의 재생 지점을 예측하여 스트림의 최초 요청 지점을 결정하므로 CoolStreaming 방안보다 재생 지점의 편차가 더 적은데다가, (그림 10)에서 보여준 결과와 같이, 평균 재생 시작 지연 또한 CoolStreaming 방안보다 더 적다.



(그림 12) 제어 오버헤드

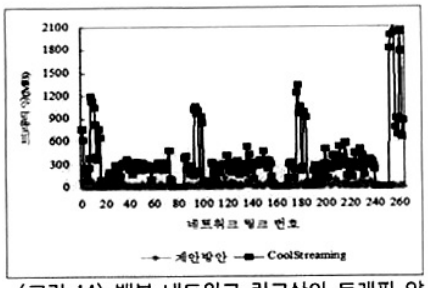


(그림 13) 소스 피어와 피어간 재생 지점의 차이

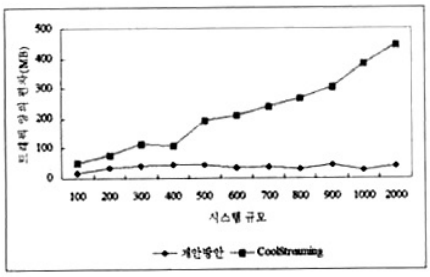
3.3 네트워크 성능 평가

네트워크 자원 활용 측면에서 성능을 평가하기 위해 각 네트워크 링크 상의 트래픽의 양과 편차를 측정해보았다. (그림 14)는 피어 1000개를 조인시켰을 때 네트워크 링크상의 트래픽 양의 합을 보인 것이다. CoolStreaming 방안에서의 트래픽 양은 평균 320MB인 반면, 제안방안에서는 평균 20MB를 보였다. 제안방안에서는 피어가 새롭게 조인했을 때 가능한 한, 즉 피어별 업로드 다운로드 대역폭 제약이 허용하는 한 동일 PoP에 있는 피어들과 통신하도록 우선적으로 피어링을 제시해주기 때문에 평균적으로 AS내 또는 AS간(254~260번 링크가 AS간 링크) 트래픽 양을 Cool- Streaming의 1/16배로 줄일 수 있었다. 제안 방안에서 AS내 또는 AS간 트래픽 양을 줄일 수 있었다는 것은 [3]에서 제시한 P4P의 목적 즉, 네트워크 사업자 입장에서의 네트워크 자원 활용 효율성 향상에 제안하는 방안이 부합됨을 보인 것이다.

(그림 15)는 네트워크 링크상의 트래픽 양의 편차를 보인 것이다. 제안방안에서는 시스템의 규모에 상관없이 네트워크 링크상의 트래픽이 고르게 분산되는 반면, CoolStreaming 방안의 경우 시스템의 규모가 커짐에 따라 트래픽 양의 편차가 증가함을 볼 수 있다. 제안방안은 P4P에 기반하고 있기 때문에 네트워크 링크상에 혼잡이나 지연이 발생하는 경로로 P2P 트래픽이 지나지 않도록 함으로써 피어의 트래픽을 제어하기 때문이다.



(그림 14) 백본 네트워크 링크상의 트래픽 양



(그림 15) 네트워크 링크상의 트래픽 양의 편차

4. 결 론

본 논문에서는 라이브 멀티미디어 스트리밍의 서비스 품질을 지원할 수 있는 P4P 기반의 P2P 시스템을 제안하였다. 제안하는 방안에서는 네트워크 제공자 측 서버인 iTracker로부터 네트워크 자원 활용 정보와 더불어 지연 및 혼잡 링크에 관한 네트워크 상태 정보를 획득하여 이 정보를 기반으로 피어링 제의를 수행하도록 하였다. 그리고 피어의 업로드/다운로드 링크 용량과 피어의 스트림 가용성 가중치를 피어링 시에 적용하도록 함으로서 지속적인 스트리밍을 지원할 수 있는 피어를 우선적으로 선택하도록 하였다. 또한 모든 피어들이 소스 피어의 재생 지점을 기준으로 일정 범위 내에서 재생을 시작하도록 하는 라이브 멀티미디어 스트리밍 재생 동기화 방안을 제안하였다. 시뮬레이션을 통해 성능을 평가한 결과, 본 방안은 네트워크상의 혼잡 링크의 발생을 줄이고, 각 링크별 트래픽 부하를 균등하게 유지하는 등 P4P의 기본 목적인 네트워크 자원 활용 문제를 효과적으로 다루면서, 라이브 멀티미디어 스트리밍에서 패킷 손실 측면에서의 요구사항을 만족하는 피어의 비율을 0.2배 향상시키고, 재생 시작 지연을 줄임을 볼 수 있었다. 또한 제어 오버헤드 1/14로 줄임으로써 시스템의 확장성을 도모할 수 있음을 확인하였으며, 시스템 전체 피어들 간 재생 지점의 편차를 최대의 경우 약 30초 줄임으로써 실시간성을 개선함을 볼 수 있었다.

참 고 문 헌

[1] Cisco, "Cisco Visual Networking Index: Forecast and Methodology, 2009~2014", Cisco White Paper, June 2010.

[2] Ipoque, "Internet Study 2008/2009," Hendrik Schulze, Klaus Mochalski, 2009. http://ipoque.com/en/pressrelease_ipoque_241006.html

[3] H. Xie, R. Yang, A. Krishnamurthy, Y. Liu, and A. Silberschatz, "P4P:Provider portal for (P2P) applications," in Proc. of ACM SIGCOMM, pp.351-362, 2008.

[4] S. Xie, B. Li, G. Y. Keung, and X. Zhang, "Coolstreaming: Design, Theory, and Practice," IEEE Transactions on multimedia, Vol. 9, No. 8, pp.1661-1671, 2007.

[5] X. Zhang, J. Liu, B. Li, and T.S.P. Yum, "CoolStreaming/DONet: A data-driven overlay network for peer-to-peer live media streaming," IEEE Infocom, pp.2102-2111, 2005.

[6] www.pplive.com

[7] A. Poli, and P. Giacomazzi, "Performance analysis of mesh-based peer-to-peer video streaming systems,", GTTI, 2010.

[8] Z. Ren, J. Liu, F. Qin, and Y. Wang, "A Survey on Peer-to-Peer Streaming System's Neighbor Number, " WCSE, pp.421-424, 2009.

[9] NS-2, www.isi.edu/nsnam/ns

[10] http://www.bandwidth.com/wiki/article/QoS_(Quality_of_Service)

[11] Steven Bauer, David Clark, William Lehr, "The Evolution of Internet Congestion," study paper, 2009

[12] R.Kumar, Y. Liu, and K. W. Ross, "Stochastic fluid theory for P2P streaming systems," in Proc. IEEE Infocom, pp. 919-927, 2007.

[13] D. Liben-nowell , H. Balakrishnan , and David Karger, "Observations on the Dynamic Evolution of Peer-to-Peer Networks," IPTPS, 2002.

변 해 선

e-mail : ladybhs@ewhain.net
2001년 광주대학교 컴퓨터학과(학사)
2003년 이화여자대학교 컴퓨터학과(공학석사)
2003년~현 재 이화여자대학교 컴퓨터학과 박사과정
관심분야 : 멀티미디어 스트리밍, P2P 네트워크, VPN, 유무선 네트워크 등

이 미 정

e-mail : lmj@ewha.ac.kr
1987년 이화여자대학교 전사계산학과(학사)
1989년 University of North Carolina at Chapel Hill 컴퓨터학과(공학석사)
1994년 North Carolina State University 컴퓨터공학과(공학박사)
1994년~현 재 이화여자대학교 컴퓨터학과 교수
관심분야 : 프로토콜 설계 및 성능 분석, 멀티미디어 전송을 위한 트래픽 제어, 인터넷 QoS, 트래픽 엔지니어링, 무선 이동 네트워크, Ad-hoc 네트워크