데이터 분할 방식의 NVOD에 패칭을 결합한 참 VOD 서비스의 제안

지용진* · 김남훈** · 박호현*

A Proposal for a True VOD Service Combining Patching with a Data Segmented NVOD

Yong-jin Ji* · Nam-hoon Kim** · Ho-hyun Park*

이 논문은 2006년 정부(교육인적자원부)의 재원으로 학술진흥재단의 지원을 받아 수행된 연구임
(사업명: 신진교수지원사업, 과제번호: D00552)

요 약

NVOD의 초기 대기시간을 없애는 방법으로 패칭이라는 방법이 있었다. 그 방법은 시간 분할의 NVOD 서비스에만 적용이 가능하였다. 그러나 데이터 분할의 NVOD 방식이 시간 분할의 NVOD 보다 대기 시간이 작거나 서버의 대역폭을 적게 사용하는 것으로 알려져 있어서, 패칭을 데이터 분할의 NVOD에 적용하면 보다 효율적인 서비스가 가능할 것이라 예상된다. 그러나 그러한 시도는 아직 없었다. 본 논문에서는 데이터 분할 NVOD에 패칭을 결합하여 시간분할 보다 더 적은 대역폭을 사용하면서 초기 대기시간을 없게 하여 참 VOD 서비스가 가능한 새로운 VOD 스케쥴링 알고리즘을 제안한다. 제안된 방식은 또한 패칭에 의한 클라이언트의 버퍼링에 필요한 스토리지 부담 요소를 NVOD 스케쥴링의 데이터 분할 정책에 흡수해 클라이언트 자원의 추가 부담 없이 참 VOD 서비스가 가능하도록 해 클라이언트의 자원 제약이 있는 경우에도 적용 가능하다.

ABSTRACT

The patching technique has been used for removing initial waiting time of NVOD. The technique was applicable only to time segmented NVOD services. However, data segmented NVOD methods have been known to have less initial waiting time or use less server's bandwidths than the time segmented NVODs. Therefore, we expect that if patching is applied to a data segmented NVOD, a more efficient NVOD service will be possible. But there has not been such approach. In this paper, we propose a new VOD scheduling algorithm that combines the patching technique with a data segmented NVOD, thus it uses less server's bandwidths and removes initial waiting time, hence makes a true VOD service possible. The proposed technique also absorbs the storage overhead necessary for client buffering incurred by patching into the data segmentation policy of the NVOD scheduling, hence does not need any extra overhead of the client's resource. Therefore, it is also applicable to the case that the client has resource limits.

키워드
VOD, 주기적 방송, 패칭, 데이터 분할

* 중앙대학교 전자전기공학부
** 동양공업전문대학 인터넷정보과

접수일자 2008. 09. 19

Ⅰ. 서 론

1.1 연구 배경

최근 VOD(Video On Demand)에 대한 서비스가 꾸준히 증가되고 있는 추세이다. 인터넷 보급의 확산, 디지털 위성 및 케이블 방송, DMB 등의 새로운 방송 방식의 확대, 방송과 통신의 융합 추세에 따른 IP-TV 등의 등장 및 휴대폰, PDA, PMP 등의 휴대 단말기 보급의 확대로 인해 VOD 서비스에 대한 수요는 나날이 증가하고 있다.

현재 서비스되고 있는 VOD는 대표적으로 RVOD (Real VOD)와 NVOD(Near VOD)가 있다. RVOD는 현재 거의 모든 방송사의 웹사이트에서 서비스되고 있다. 이는 사용자가 원하는 시간에 클릭 한번 만으로 비디오를 선택하여 볼 수 있기 때문에 사용자의 입장에서는 가장 편한 방법이다.

NVOD는 현재 호텔, 케이블 방송, 위성 방송에서 서비스되고 있다. NVOD는 여러 채널을 통하여 주기적으로 반복 전송하고 하나의 채널에 다수의 사용자들이 접속할 수 있는 브로드캐스팅(broadcasting) 또는 멀티캐스팅(multicasting) 방식을 이용하여 서버의 비용을 획기적으로 줄인 방식이다. 단, NVOD 방식에서 사용자는 대화형 서비스를 포기하고, 비디오를 시청하기 전에 사용자가 접속한 채널로 스트림이 전송되기 시작하는 시점까지 기다리는 시간(access latency)을 감수해야 한다.

RVOD는 사용자가 원하는 시간에 비디오를 선택하여 본다는 장점이 있지만 하나의 사용자가 서버로부터 단말까지 하나의 전송채널을 점유하기 때문에 사용자가 많아지면 서버 및 네트워크 폭주 현상, 폭주로 인한 전송 지연 및 끊김 현상이 발생할 수 있다. 문제를 해결하기 위해서는 서버 용량과 네트워크 대역폭을 무한정 늘려야 하므로 이는 엄청난 서버 및 네트워크 구축 비용을 필요로 한다. 즉, RVOD는 확장 가능성(scalability) 관점에서 볼 때 사용자가 많은 경우에는 적합한 해결책이 되지 못한다.

NVOD에서 서버의 채널을 할당하는 방식은 크게 시간을 분할하는 방식과 데이터를 분할하는 방식으로 나눌 수 있다. 시간을 분할하는 방식은 현재 호텔, 케이블 방송, 위성 방송 등에서 많이 상용 서비스 하고 있는 재래식 방송(Conventional Broadcasting) 방식이 있다 [1]. 그러나 이 방식은 사용자 대기 시간이 너무 긴 단점이 있다. 현재의 Skylife의 NVOD 서비스인 SkyChoice의 경우에 인기 비디오에 대해 30분의 대기시간을 필요로 한다.

이러한 대기시간 문제를 해결하기 위해 피라미드 방송(Pyramid Broadcasting)[2], 순열 기반 피라미드 방송(Permutation-based Pyramid Broadcasting)[3], 마천루 방송(Skyscraper Broadcasting)[4], 조화 방송(Harmonic Broadcasting)[5], 피보나치 방송(Fibonacci Broadcasting) [6] 등의 데이터 분할 방식이 개발되었다. 이들 방식은 서버 측에 요구되는 대역폭의 감소와 사용자의 대기시간의 감소에 초점을 둔다. 그러나 이들 방법 대부분은 서버의 채널 대역폭과 사용자의 대기시간을 줄이는 대신에 클라이언트에 많은 대역폭과 스토리지를 부담하는 단점이 있다. 때문에 클라이언트를 고려하여 개발된 유연 방송(Flexible Broadcasting) 방법이 있다[7]. 이 방법은 기존의 분할 방식들보다 대기시간, 서버대역폭, 클라이언트 스토리지 부담 측면에서 우수하다.

1.2 연구 동기 및 목적

데이터 분할 방식들은 기존의 시간 분할 방식에 비해 대기 시간을 많이 감소 시켰지만 여전히 NVOD의 한계인 초기 사용자 대기 시간은 존재한다. 이러한 초기 대기 시간은 모든 NVOD 서비스, 주기 방송(Periodic Broadcasting) 또는 멀티캐스팅 방식을 사용하는 어떠한 형태의 스트리밍에서도 나타나는 현상이다.

주기 방송 또는 멀티캐스팅 방식에서 초기 대기 시간 없이 참(true) VOD 서비스가 가능하도록 멀티캐스팅과 유니캐스팅을 적절히 조합하는 패칭(patching)이라는 방식이 제안되었다[8-12].

기존의 NVOD 방식들에서는 사용자가 VOD 서비스 접속 요구 시 비디오 시작 부분의 멀티캐스트 스트림이 이미 지나가 버렸을 때, 비디오의 시작 부분이 다시 전송되기까지 기다려야 하지만, 패칭은 바로 그 시점부터 멀티캐스트 스트림을 다운 받아 클라이언트에 버퍼링을 함과 동시에 이미 놓친 부분은 유니캐스트로 별도의 채널로 다운 받아 플레이하고 유니캐스트가 끝나면 멀티캐스트 채널에서 버퍼링 해 놓은 부분을 플레이 하면서 멀티캐스트 스트림을 일정 오프셋을 두고 쫓아 가는 방식이다. 패칭을 위해서 클라이언

트는 유니캐스트를 위한 별도의 네트워크 채널과 초기에 놓친 데이터 량 만큼의 버퍼를 추가로 확보해야 한다.

시간 분할 방식(Conventional Broadcasting)의 NVOD 서비스에 패칭을 결합한 기법들은 기존에 연구되어왔다[8,10]. 그러나 시간 분할 방식 보다 사용자 초기 대기시간이 작고 서버의 대역폭 사용량도 적은 데이터 분할 방식에 패칭을 결합하여 참 VOD를 실현하고자 하는 시도는 아직 없었다. 패칭을 시간 분할 방식 보다 효율이 좋은 데이터 분할 방식에 적용하면 보다 적은 서버 대역폭을 가지고 참 VOD 서비스가 가능할 수 있을 것으로 예상된다. 그러나 데이터 분할 방식에 패칭을 결합하는 문제는 클라이언트에서 비디오가 끊김 없이 플레이가 가능해야 한다는 관점에서 볼 때 시간 분할 방식과 패칭의 결합만큼 간단하지는 않다.

본 논문에서는 데이터 분할 방식에 패칭을 끊김없이 결합하기 위한 방법을 연구하고, 데이터 분할 방식의 NVOD 서비스와 패칭 기법과의 결합이 시간 분할방식과의 결합보다 서버 대역 효율이 우수함을 보인다. 본 논문에서 제안하는 데이터 분할 NVOD에 패칭을 결합하는 알고리즘은 클라이언트 대기시간을 없게 하여 참 VOD 서비스가 가능하도록 한다. 또한 제안된 방식은 패칭에 의한 클라이언트의 버퍼링에 필요한 스토리지 부담 요소를 NVOD 스케쥴링의 데이터 분할 정책에 흡수해 클라이언트 자원의 추가 부담 없이 참 VOD 서비스가 가능하도록 해 클라이언트의 자원 제약이 있는 경우에도 적용 가능하다.

1.3 관련 연구

1) 재래식 방송 (CB: Conventional Broadcasting)

RVOD는 사용자의 요청에 따라 엄청난 서버의 대역폭을 요구하기 때문에 이런 단점을 개선하기 위해 많은 NVOD서비스 방법들이 생각되었다 그중에서 전통적으로 많이 이용되고 있는 NVOD서비스 방법은 재래식 방송(Conventional Broadcasting)이다[1]. 이 방법은 전체 비디오를 일정한 간격을 두고 여러 채널을 통해 전송하여 다수의 가입자가 동시에 비디오를 시청할 수 있도록 하는 방법이다. 예를 들어 120분 길이의 비디오를 12개의 채널을 통해 전송하면 각 채널의 시간 간격은 10분이 되므로 대기시간은 10분이 된다. 이 방법의 장점은 단말기기에 저장공간이 필요 없다는 점이고 단점은 서버의 대역폭에 비해 너무 큰 대기시간을 갖는다는 점이다.

2) 피라미드 방송 (PB: Pyramid Broadcasting)

이 방법은 비디오 데이터를 분할해서 전송한 최초의 방법이다. 비디오 데이터를 여러 부분으로 분할하고 여러 채널을 이용하여 분할된 데이터를 전송한다[2]. 만약 비디오를 k개로 분할하였다면 k개의 채널이 필요하게 된다. 데이터조각의 크기는 α^i(i=1, 2, 3…k)이고 $\alpha \geq 1$ 이다. α를 2라고 하면 1, 2, 4, 8, 16, 32...의 비율로 데이터를 분할하고 k개의 채널에 각 데이터 조각들은 반복하여 전송한다. 사용자는 각 채널을 순서대로 이동해 가면서 전송 받는다. 예를 들어 120분 길이의 비디오를 12b(b:비디오의 재생 비트율)의 채널로 전송할 경우 대기시간은 1분 이내로 줄어들게 된다. 이는 재래식 방송에 비해 90%이상의 대기시간 감소효과를 나타내는 것이다. 이 말은 바꿔 생각하면 같은 대기시간을 가질 경우 상당히 작은 대역이 필요하다는 말이 될 수 있다. 하지만 비디오의 50%이상을 저장할 수 있는 저장공간이 필요하다는 것이 큰 단점이다. 그리고 단말은 2 ab의 대역폭을 필요로 한다.

3) 유연 방송 (FB: Flexible Broadcasting)

유연 방송은 클라이언트 중심적 접근을 하여 클라이언트의 네트워크 대역폭과 저장 공간에 따라 유연하게 적응 가능한 방법이다[7]. 피라미드 방송과 같은 기존의 방법과 다른 데이터의 분할을 사용함으로서 서버 대역 효율을 증가시켰고, 클라이언트 네트워크 대역폭의 크기에 따라 분할 방식이 바뀌게 되고, 클라이언트의 버퍼 크기를 고려하여 버퍼 크기가 작은 클라이언트도 서비스 받을 수 있도록 분할됨으로서 피라미드 방송의 가장 큰 단점이었던 클라이언트 버퍼 부담을 해소하였다.

처음 m개 분할은 2i-1의 비율로 분할한다. 만약 m=3이면 앞의 세 분할의 비율은 {1, 2, 4} 의 비율로 분할된다. m개 분할 이후의 분할 방법은 해당 분할 이전의 m개의 분할을 합한 크기로 분할한다. 예를 들어 m=3일 때 앞 세 분할의 크기의 합은 1+2+4 이므로 네 번째 분할의 크기는 7이 되고 5번째 분할의 크기는 2+4+7 이므로 13이 된다. 클라이언트의 저장 공간이 Smax로 제한되었다면 보다 크게 분할되어야 하는 경우에도 해당 분할의 크기

를 Smax로 강제한다.

유연 방송은 기존의 피라미드 방송에 비해 서버의 부담이 적어졌을 뿐 아니라, 클라이언트 측면에서 피라미드 방송의 가장 큰 문제점이었던 클라이언트의 스토리지 부담이 해결되었고 클라이언트 요구 대역폭도 줄게 되었다.

4) 패칭 (Patching)

패칭은 동일한 비디오를 요청하는 클라이언트 사이에서 하나의 패칭 윈도우 안에 임의의 시간차로 도착하는 클라이언트들이 비디오의 앞부분인 시작 스트림을 재생해보는 동안 멀티캐스트 되는 정규 스트림을 클라이언트 버퍼에 담아두었다가 패칭 채널을 통해 유니캐스트로 전송하는 방식이다[8-12]. 패칭을 하기 위해서는 정규 멀티캐스트 채널을 통해 스트림을 받을수 있는 만큼의 클라이언트 대역폭과, 놓친 만큼의 스트림을 패칭 채널을 통하여 받을 수 있는 만큼의 추가적인 클라이언트 대역폭, 채널을 받는 동안 정규 스트림을 저장할 수 있는 클라이언트의 버퍼 용량이 요구된다. 패칭 방법의 가장 큰 장점은 대기시간이 전혀 없는 참 VOD서비스를 가능케 한다는 점과 멀티캐스팅을 이용해 다수의 클라이언트에게 서비스하기 때문에 서버의 부하를 감소시킨다는 점이다.

Ⅱ. 패칭을 결합한 데이터 분할 방식의 NVOD

시간 분할 방식의 NVOD에 패칭을 결합한 방법들은 기존에 연구되었지만, 시간 분할 방식에 비해 서버 대역 효율이 우수한 데이터 분할 방식에의 결합은 연구되지 않았다.

본 논문에서는 시간 분할 방식보다 서버의 효율이 좋은 데이터 분할 방식에 패칭을 결합하기 위한 알고리듬을 연구하고, 시간 분할 방식에 패칭을 적용한 경우와 비교해 본다.

2.1 파라미터 정의

비디오 데이터의 분할, 서버의 전송 대역폭과 클라이언트 대역폭, 클라이언트의 버퍼 크기 등을 나타내기 위해 다음과 같은 파라미터를 사용한다.

S : 전체 비디오의 길이 (단위 : sec)

k : 분할의 개수 = 채널의 개수

s_i : 비디오의 i번째 분할의 길이 (단위 : sec)

b : 비디오의 재생 대역폭 (단위 : Mbps)

B : 서버에 필요한 전송 대역폭 (단위 : Mbps)

B_p: 서버에 필요한 패칭 채널의 평균 전송 대역폭 (단위 : Mbps)

b_i : i번째 채널의 대역폭 (단위 : Mbps)

m : b에 대한 단말의 상대적인 네트워크 대역폭의 크기

$S_{\max}$: 클라이언트 버퍼 크기 (단위 : MB(Mega Byte))

N : 서비스 중인 사용자 수

N_p: 패칭 중인 사용자 수

w : 최대 대기 시간의 길이 (단위 : sec)

t : 비디오 정규 채널의 시작 시점부터 지난 시간 (단위 : sec)

2.2 PB, FB에의 패칭 적용

데이터 분할의 NVOD에 패칭을 적용시킬 경우 끊김 없이 재생이 되기 위해서는 다음의 3가지 조건을 만족해야 한다.

① 첫 번째 분할의 재생이 끝나기 전까지 두 번째 세그먼트의 시작 부분이 버퍼링되어 있어야 한다. 그러기 위해서는 S_2의 전송 주기가 S_1 재생 시간보다 짧거나 같아야 S_2가 재생되기 전에 따라잡을 수 있다.

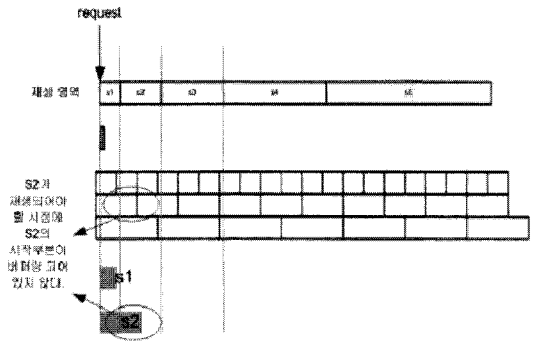

그림 1. 패칭을 결합하기 위한 조건-1
Fig 1. Condition 1 to combine patching

위의 그림은 요청시점이 12초이고, 각 분할의 재생 시간과 전송 주기는 S_1=1분, S_2=2분, S_3=3분이다.

S_1이 재생되고 있는 동안에 S_2를 버퍼링하는데 S_1의 재생 시간이 S_2의 전송 주기보다 짧으면, 클라이언트는 S_2가 재생되어야 할 시간에 S_2의 앞부분을 받을 수 없다.

② 요청 시점 이전 부분은 패칭 채널을 통해 전송하고, 이후의 부분은 버퍼에 담아 두었다가 재생을 한다. 따라서 이후의 부분을 버퍼에 담기 위해서는 각 분할의 시작 부분이 아니어도 버퍼링이 가능해야 한다.

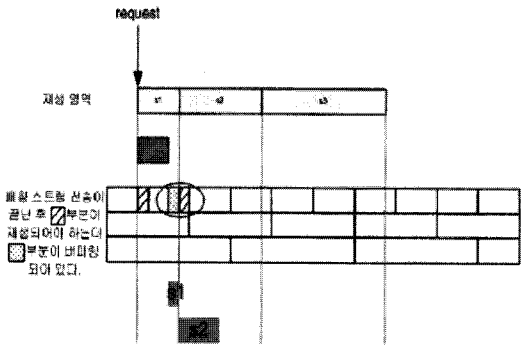

그림 2. 패칭을 결합하기 위한 조건-2
Fig 2. Condition 2 to combine patching

③ 패칭 채널이 재생되는 동안 버퍼에 다음 부분을 담아둬야 하기 때문에 최소 동시에 2개 이상의 채널을 받을 수 있는 대역폭이 필요하다. 동시에 1개의 채널만 받을 수 있는 경우에는 그림 2에서처럼 놓친 부분의 바로 다음 부분이 버퍼링되지 못하여 끊김 현상이 발생할 수 있다.

위의 조건을 만족하는지 피라미드 방송과 유연 방송에 각각 패칭을 적용시켜 본다.

1) 패칭을 결합한 피라미드 방송

그림 3은 α=2인 피라미드 방송에 패칭을 적용한 경우이다. 피라미드 방송에서 m값은 사용되지 않지만, 첫 번째 분할이 전송되기 시작한 시점부터 최대한 빨리 다음 분할을 버퍼링하기 때문에 재생대역폭 b의 상대적인 크기인 클라이언트의 네트워크 대역폭 m을 동시에 2개의 채널을 받을 수 있다는 의미로 4라고 한다.

S_2의 전송 주기가 S_1 재생 시간보다 짧거나 같아야 S_2가 재생되기 전에 따라잡을 수 있다.

피라미드 방송에서는 첫 번째 채널의 재생 시간과 두 번째 채널의 전송주기가 같기 때문에 이 부분에 문제가 없다.

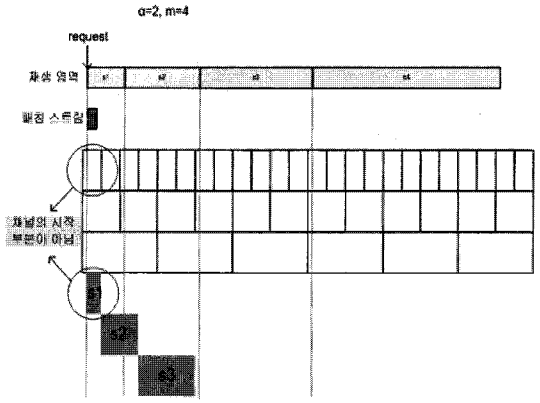

그림 3. 패칭을 결합한 피라미드 방송
Fig 3. PB is combined with patching

하지만, 놓친 부분 이후의 부분이 재생이 되기 시작한 시점부터 바로 버퍼링 되어야 하는데 피라미드 방송의 경우 채널의 시작 부분부터 버퍼링이 이루어져야 하므로 ② 조건을 만족시키지 못한다.

2) 패칭을 결합한 유연 방송

유연 방송은 버퍼에 저장되는 스트림이 각 채널이 재생될 때까지 시작 부분을 버퍼에 받아야만 끊김 없이 플레이가 가능하다. 하지만 유연 방송은 S_2의 전송 주기가 S_1보다 길다.

그림 4는 m=2인 유연 방송에 패칭을 적용시킨 경우이다. 요청 시점부터 패칭 채널을 통해 바로 플레이가 된다. 하지만 2번째 세그먼트가 재생되어야 할 부분에서 아직 2번째 세그먼트 시작 부분을 버퍼링하지 못한다. 따라서 시작 부분을 받을 때까지 끊김 현상이 발생하게 된다.

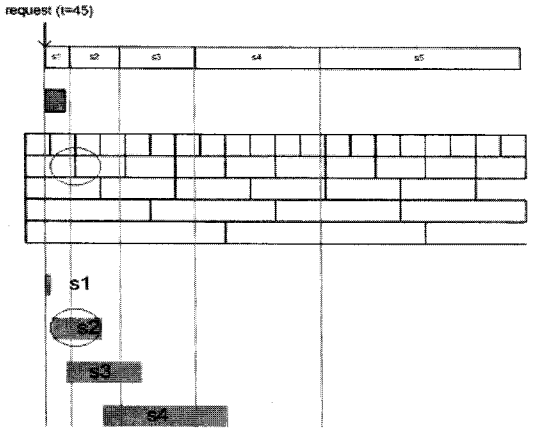

그림 4. 패칭을 결합한 유연 방송
Fig 4. FB is combined with patching

유연 방송의 경우는 ① 조건을 만족시키지 못한다.

① 조건을 만족시키기 위해서는 그림 5와 같이 2번째 분할까지는 무조건 패칭 채널을 통해 전송하는 방법이 있다.

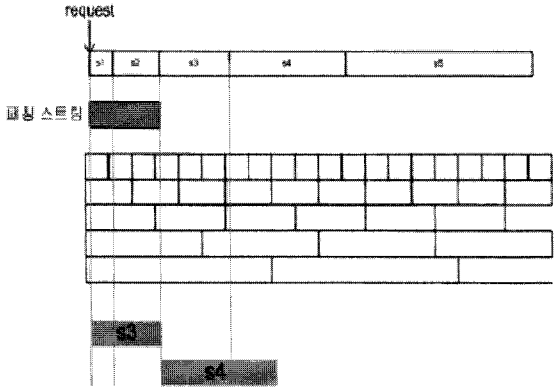

그림 5. 패칭을 결합한 유연 방송
(2번째 세그먼트까지 패칭)
Fig 5. FB is combined with patching
(patching until the 2nd segment)

하지만 이 경우는 패칭 스트림의 길이가 S_1+S_2로 고정되기 때문에 패칭 채널의 존속 시간이 최대 S_1일 때보다 최소 3배 이상 길어지게 된다. 따라서 서버 대역폭이 증가하게 된다.

2.3 패칭 보조 채널을 사용한 **FB**에 패칭 결합

① 조건을 만족시키기 위해서 유연 방송은 2번째 채널까지 무조건 패칭을 해야하는 문제가 있다. 따라서, ① 조건을 만족시키면서 2번째 채널까지의 패칭을 피할 수 있는 방법으로 패칭 보조 채널을 사용한다. 이 방법은 첫 번째 분할의 크기가 1분일 경우 1분짜리 채널 하나를 가장 앞 부분에 열어주는 방식이다.

예)

FB(m=2, 60분 비디오) :

패칭보조채널=1분 S_1=1분 S_2=2분 S_3=3분 S_4=5분 S_5=8분 S_6=13분 S_7=21분 S_8=6분

FB(m=3, 60분 비디오) :

패칭보조채널=1분 S_1=1분 S_2=2분 S_3=4분 S_4=7분 S_5=13분 S_6=24분 S_7=8분

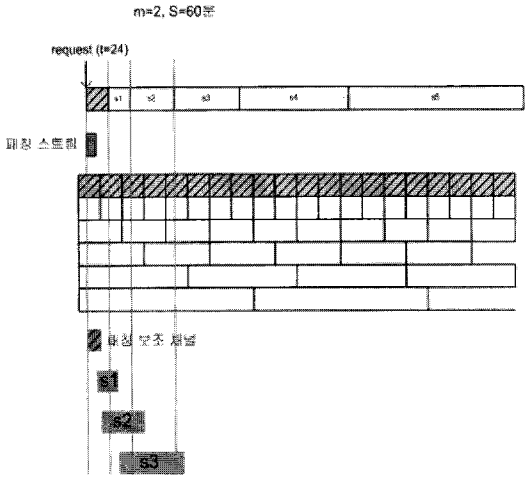

그림 6. 패칭 보조 채널을 사용한 유연 방송에 패칭 적용
Fig 6. Apply patching to FB that uses the patching auxiliary channel

이 방법을 쓰게 되면, 첫 번째 채널의 길이가 1분이 되고 두 번째 채널의 크기도 1분이 되므로, 기존에 S_2의 전송 주기가 S_1 재생 시간보다 길어서 S_2가 재생되기 전에 따라잡을 수 없었던 문제를 해결할 수 있다.

그림 6과 같이 패칭 보조 채널의 추가로 인해 S_2의 전송 주기가 S_1 재생 시간보다 짧거나 같아져 S_2가 재생되

기 전에 따라잡을 수 없었던 문제가 해결되었고, 이로 인해 2번째 채널까지는 무조건 패칭 채널을 통해 전송해야 할 필요가 없어졌기 때문에 패칭 채널의 길이가 1/3 이하로 감소하게 된다.

2.4 평균 서버 대역폭

FB에 패칭을 적용한 경우의 서버대역폭 B는 정규 채널의 크기에 패칭보조채널 1개와 패칭 채널의 크기 B_p를 더한 값이다. B_p의 크기는 패칭 채널이 열린 시간 $t_0, t_1, t_2, \ldots, t_i$에 패칭 중인 사용자 수 N_{p_i}를 곱하고 재생대역폭을 곱한 후, 비디오의 길이 S로 나눈 값이 된다.

$$B = (k+1) \times b + B_p$$
$$B_p = \frac{b \sum N_{p_i} \times t_i}{S}$$

패칭 채널이 열린 시간 t_i는 첫 번째 분할 내에서 요청 시점부터 다음 스트림의 시작 부분까지의 시간이다. t_i는 각각의 시간 t를 패칭 채널의 최대 길이로 나눈 값과 같다. CB에 패칭을 적용했을 때와 식은 비슷하지만, CB의 경우 같은 채널 수를 사용했을 경우 FB에 비해 각각의 채널의 길이가 훨씬 길기 때문에 그만큼 패칭 채널의 길이가 길어진다. 최대 패칭 채널의 길이를 갖게 할 경우엔 정규 채널의 개수가 많아진다. 따라서 평균 서버대역폭도 더 증가하게 된다.

Ⅲ. 성능분석 및 비교

3.1 각 방식들의 서버 대역폭

각각의 방식들의 서버대역폭에 대해 살펴본다. RVOD, NVOD의 경우에는 서버대역폭의 크기가 일정하지만, 패칭을 적용한 NVOD의 경우는 각각의 클라이언트 요청 시점에 따라 패칭 채널의 존속 시간이 변하게 되기 때문에 서버대역폭이 수시로 변하게 된다. 따라서 수치적으로 확인하기는 힘들기 때문에, 패칭을 적용한 NVOD에는 평균 서버대역폭을 적용한다.

순간적인 서버대역폭은 다음 절에서 그래프를 통해 확인해보겠다.

1) RVOD

RVOD는 유니캐스트 방식으로 1명의 클라이언트가 1개의 채널을 점유하기 때문에 클라이언트의 수만큼 채널을 열어줘야 한다. 따라서 RVOD의 서버대역폭 B는 클라이언트 수에 채널의 대역폭 크기를 곱한 값이 된다.

$$B = N \times b$$

2) 재래식 방송

NVOD 방식에서는 멀티캐스팅을 사용하기 때문에 각각의 클라이언트마다 채널이 필요하지 않다. 재래식 방송 방식의 서버대역폭 B는 채널의 개수와 대역폭을 곱한 값이 된다. 최대 대기시간 w는 비디오의 길이를 채널의 개수로 나눈 값이 된다.

$$B = k \times b$$
$$w = S \div k$$

Ex) 비디오 길이가 60분이고 6개의 채널로 전송하면 최대 대기시간은 10분이 된다.

3) 유연 방송

유연 방송 방식의 서버대역폭 B는 정규 채널에 대한 대역폭만 필요하므로 채널의 개수와 대역폭을 곱한 값이 되고, 최대 대기시간 w는 첫 번째 분할의 크기가 된다.

$$B = k \times b$$
$$W = S1$$

4) 패칭을 적용한 재래식 방송

재래식 방송 방식에 패칭을 적용한 경우의 서버대역폭 B는 기존의 정규 채널에 평균 패칭 채널 대역폭 B_p만큼 더해진다. B_p의 크기는 패칭 채널이 열린 시간 $t_0, t_1, t_2, \ldots, t_i$에 패칭 중인 사용자 수 N_{p_i}를 곱하고, 재생대역폭을 곱하고 비디오의 길이 S로 나눈 값이 된다. 패칭 채널이 열린 시간 t_i는 요청 시점부터

다음 정규스트림 시작 부분까지의 시간이다. t_i는 각각의 시간 t를 최대 대기시간 w으로 나눈 나머지와 같다.

따라서

$$B = k \times b + B_p$$
$$B_p = \frac{b \sum N_{p_i} \times t_i}{S}$$

가 된다.

5) 패칭을 적용한 유연 방송

3.4절에서와 같이

$$B = (k+1) \times b + B_p$$
$$B_p = \frac{b \sum N_{p_i} \times t_i}{S}$$

가 된다.

3.2 서버 대역폭 사용 예

일정한 사용자 수를 임의적으로 만들어 패칭을 적용한 CB, FB의 평균 서버대역폭을 비교해본다.

사용자의 수가 다음과 같을 때, 각각의 방식의 평균 서버대역폭을 비교해본다. CB와 FB가 같은 정규 채널 수를 사용하고 있을 때와, 최대 패칭 채널의 길이를 갖게 한 경우의 예이다.

30분 짜리 비디오를 가정하자. CB로 6개 채널의 경우와 30개 채널의 경우를 가정하자. 이 경우 최대 초기 대기 시간 w는 각각 5분과 1분이다. FB로 6개 채널을 가정하고 데이터 분할이 1,2,3,5,8,11 분의 비율로 분할된다고 하자. 이 경우 w는 FB의 데이터 분할 정책에 의해 1분이 된다. 그리고 b=1Mbps이고, t=20에 10명, t=40에 10명, t=100에 10명, t=250에 10명, t=450에 10명, t=650에 10명, t=800에 10명, t=1000에 10명, t=1300에 10명, t=1600에 10명 씩 사용자 요청이 있었을 때,

RVOD의 서버대역폭은

$$B = N \times b$$

이고, 계산하면 B = 100×1 = 100Mbps가 된다.

CB의 서버대역폭은

$$B = k \times b$$

이고, 6채널인 경우 최대 대기시간 w는 5분이고 B = 6×1 = 6Mbps가 된다. 30채널인 경우 w는 1분이고 B = 30×1 = 30Mbps가 된다.

FB의 서버대역폭은

$$B = k \times b$$

이고, 계산하면 w는 1분이고 B = 6×1 = 6Mbps가 된다.

CB에 패칭을 적용한 경우의 평균 서버 대역폭은

$$B = k \times b + B_p$$
$$B_p = \frac{b \sum N_{p_i} \times t_i}{S}$$

이고, 6채널인 경우 B = 6×1+(20×10+40×10+100×10+250×10+150×10+50×10+200×10+100×10+100×10+100×10)/1800 = 12.17Mbps가 된다. 30채널인 경우의 B = 30×1+(20×10+40×10+40×10+10×10+30×10+50×10+20×10+40×10+40×10+40×10)/1800 = 31.83Mbps가 된다.

FB에 패칭을 적용한 경우의 평균 서버 대역폭은

$$B = (k+1) \times b + B_p$$
$$B_p = \frac{b \sum N_{p_i} \times t_i}{S}$$

이고, 계산해보면 B = (6+1) × 1 + (20×10+40×10+40×10+10×10+30×10+50×10+20×10+40×10+40×10+40×10)/1800 = 8.83Mbps가 된다.

위의 예와 같이 30분 동안 100명의 사용자가 요청을 하는 경우, RVOD는 대기시간이 없다는 장점 대신 1명의 클라이언트가 1개의 채널을 점유하기 때문에 매우 큰 서버대역폭을 갖는다. NVOD는 일정한 대기시간을 갖는다는 단점을 가지고 있지만, RVOD에 비해 매우 작은 서버대역폭을 사용한다.

패칭을 적용한 NVOD의 경우, 대기 시간이 없다는 RVOD의 장점을 살리면서 멀티캐스팅을 이용해 서버

대역폭을 줄인 NVOD의 장점까지 살렸다. 단적인 예지만 위의 예에서는, FB가 6Mbps를 사용하고 FB에 패칭을 적용한 경우에 평균 8.83Mbps의 대역폭을 사용한다. RVOD가 대기시간을 없애기 위해서 94Mbps를 더 사용한 반면, 패칭을 적용한 FB는 평균 2.83Mbps만을 더 사용함으로써 대기시간이 없는 참 VOD를 가능케 한다.

또한, 시간 분할 방식의 NVOD인 CB에 패칭을 적용한 경우와 데이터 분할 방식의 FB에 적용한 경우에 서버 대역폭의 차이가 난다. 정규 채널 수를 같게 했을 때와 최대 패칭 채널의 길이를 같게 했을 경우 모두 CB의 경우가 FB에 비해 훨씬 큰 서버대역폭을 요구한다.

3.3 순간 서버 대역폭 비교

3.1, 3.2절에서 패칭을 적용한 CB, FB의 평균 서버 대역폭을 보았다면 이번 절에서는 그래프를 통하여 순간 서버 대역폭을 비교해본다.

그림 7은 RVOD, CB와 패칭보조채널을 사용한 FB에 각각 패칭을 적용시켰을 때의 시간에 따른 대역폭의 비교이다. 30분 길이의 비디오이고, 재생대역폭 b=1Mbps이다.

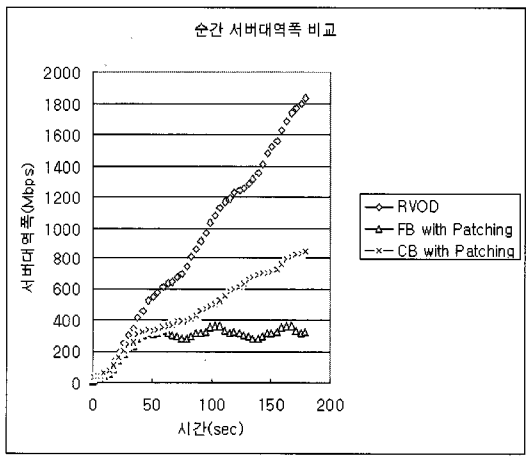

그림 7. 순간 서버 대역폭의 비교
Fig 7. Comparison of instant server bandwidth

이 비디오에 대해서 클라이언트 수를 rand() 함수를 사용하여 초당 0~20명까지 랜덤하게 분포시켜 RVOD, CB, m값이 2인 FB, 패칭을 적용한 CB, FB에 대해 비교해 본다. CB의 경우, FB와 같은 수의 정규 채널을 사용했을 때(6 채널)의 경우를 비교한다.

그림 7은 처음 3분 동안의 순간 서버 대역폭이다. RVOD는 사용자의 수에 따라 선형적으로 증가하고, 패칭을 적용한 FB의 경우는 처음에는 증가하다가 최대 패칭 채널의 길이만큼 지난 시점에서는 일정한 값을 기준으로 진동하는 형태가 된다.

Ⅳ. 결 론

VOD 서비스가 증가하면서 사용자들은 기다리지 않고 바로 볼 수 있는 VOD를 원한다. 이러한 사용자들의 요구를 충족시키기 위해서는 서버에 큰 부담이 따를 수 밖에 없었다. 현재 서비스되고 있는 RVOD가 대표적인 예이다. 사용자의 수만큼 서버의 부담이 커지므로 다수의 사용자에게 서비스 되기에는 무리가 있다. 따라서, 패칭이라는 기법을 통해 이 문제를 해결하고자 시간 분할 방식의 NVOD에 패칭을 적용하는 방법이 연구되었다. 패칭을 적용한 NVOD는 사용자의 요구는 가장 잘 충족시켜주지만 서버의 효율이 가장 안 좋은 유니캐스트 방식의 RVOD와, 일정 대기 시간이 존재하지만 서버의 효율 면에서는 가장 유리한 멀티캐스트 방식의 NVOD 각각의 장점을 살릴 수 있는 유니캐스트와 멀티캐스트의 조합이다.

본 논문에서는 기존에 연구되었던 시간 분할 방식보다 서버의 효율이 좋은 데이터 분할 방식에 패칭을 적용시키는 방법을 연구하였다. 본 연구에 사용된 데이터 분할 방식의 NVOD 스케쥴링 방식으로는 유연 방송 알고리즘을 이용하였다. 시간 분할 방식에 패칭하는 것과는 달리 데이터 분할 방식에 패칭을 적용하는 문제는 그리 간단하지 않았다. 본 논문에서는 이 문제를 패칭 보조 채널을 이용함으로써 해결하였다. 데이터 분할 방식의 NVOD에 패칭을 결합한 결과, 시간 분할 방식의 NVOD 경우보다 서버 효율이 좋다는 것을 알 수 있었다.

기존의 RVOD와 시간분할 NVOD에의 패칭 이외에, 서버의 효율을 좀 더 높일 수 있는 패칭을 적용한 데이터 분할 방식의 유연 방송 알고리즘은 증가하는 VOD 서비스 요구 속에서 유용한 서비스가 될 수 있을 것이다.

참고문헌

[1] K. c. Almeroth and M. H. Ammar. " On the use of multicast delivery to provide a scalable and interactive video-on-demand service," IEEE Journal on Selected Areas in Communications, 14(5):1110-22, Aug 1996.

[2] S. Viswanathan and T. Imielinski. "Pyramid Broadcasting for video on demand service," In IEEE Multimedia Computing and Networking Conference, Volume 2417, pp 66-77, San Jose, California, 1995.

[3] C. C. Aggarawal, J. L. Wolf, and P. S. Yu. "A permutation-based pyramid broadcasting scheme for video-on-demand systems," In Proc. of the IEEE Int'l conf. of Multimedia Computing and Systems '96, Hiroshima, Japan, June 1996

[4] K. A. Hua and S. Sheu. Skyscraper Broadcasting: a new broadcasting scheme for metropolitan video-on-demand systems. In ACM SIGCOMM 97, pp 89-100, Cannes, France, Sept. 1997.

[5] L. Juhn and L. Tseng. Harmonic broadcasting for video-on-demand service. IEEE Transactions on Broadcasting, 43(3):268-271, Sept. 1997.

[6] Ailan Hu. Video-on-Demand Broadcasting Protocols: A Comprehensive Study, IEEE Infocom 2001, pp 508-517, April 2001.

[7] 정민성, 박호현, 황재식, "다양한 단말환경에 적응되는 NVOD 스케쥴링 방법과 서비스," Telecommunications review, 제16권 제2호, pp.228-245, April 2006.

[8] K. Hua, Y. Cai, and S. Sheu. "Patching: A multicast technique for true video-on-demand services," In Proc. ACM Multimedia, September 1998.

[9] Y. Cai, K .Hua and K. Vu, "Optimizing Patching Performance," In Proc. SPIE/ACM Conference on Multimedia Computing and Networking, pp204-215, January 1999

[10] S. Sen, L, Gao, J. Rexford, and D. Towsley, "Optimal patching schemes for efficient multimedia streaming," In Proc. NOSSDAV'99, Basking Ridge, NJ, June 1999.

[11] Huadong Ma, Kang G. Shin, and Weibiao Wu, "Best-Effort Patching for Multicast True VoD Service," Multimedia Tools Applications, 26(1):101-122, May 2005.

[12] P. P. White and J. Crowcroft, "Optimized batch patching with classes of service," ACM communications. Rev., vol. 30, no. 4, Oct. 2000.

[13] Michael K. Bradshaw, Bing Wang, Subhabrata Sen, Lixin Gao, Jim Kurose, Prashant Shenoy, and Don Towsley "Periodic Broadcast and Patching Service - Implementation, Measurement, and Analysis in an Internet Streaming Video Testbed," Proc. ACM Multimedia'2001, Ottawa, Canada. 2001.

저자 소개

지용진 (Yong-Jin Ji)

2006년 2월 중앙대학교 전자전기공학부 (학사)
2007년 9월 ~ 현재 중앙대학교 전자전기공학부 (공학석사)
※관심분야 : 멀티미디어 시스템, 데이터베이스 시스템

김남훈 Nam-hoon Kim)

1988년 2월 연세대학교 전산과학과 (이학사)
1996년 8월 한국과학기술원 정보통신공학과 (공학석사)
2006년 8월 세종대학교 컴퓨터공학과 (공학박사)
1988년 1월 ~ 1998년 2월 삼성SDS 정보기술연구소 (선임연구원)
1998년 3월 ~ 현재 동양공업전문대학 인터넷정보과 (부교수)
※관심분야 : 멀티미디어 데이터베이스, 정보통신, 웹 응용

박호현 (Ho-hyun Park)

1987년 2월 서울대학교
계산통계학과 (이학사)
1995년 8월 한국과학기술원
정보통신공학과 (공학석사)
2001년 2월 한국과학기술원 전자전산학과 (공학박사)
1987년 1월 ~ 2003년 2월 삼성전자 정보통신총괄
(수석연구원)
2003년 3월 ~ 현재 중앙대학교 전자전기공학부
(부교수)
※관심분야 : 멀티미디어 스트리밍, 멀티미디어 검색,
유비쿼터스