Smooth 스트리밍을 이용한 클라이언트 제어

강현규* 문영성
숭실대학교 컴퓨터학과
hkkang@sunny.ssu.ac.kr, mun@computing.ssu.ac.kr

Control of Client Using Smooth Streaming

Hyunkyu Kang*, Youngsong Mun
School of Computing, Soongsil University

요 약

인터넷을 통한 실시간 멀티미디어 전송에 있어서 인터넷이 "Best effort"서비스이기 때문에 지연과 지연변이 그리고 패킷 손실 등이 발생하게 되어 서비스의 질에 큰 영향을 미치게 된다. 더욱이 멀티미디어 데이터는 시간에 민감하다는 속성 때문에 네트워크상의 지연이나 지연 변이는 클라이언트 측의 재생지연과 버퍼의 Underflow, 미디어 간 동기의 어긋남을 발생시킬 수 있다.

본 논문에서는 이를 해결하기 위해서 네트워크나 서버 측에서의 접근이 아닌 클라이언트의 측면에서 접근하였다. 기존 adaptive한 기법[2][3]을 도입, 개선하므로써 클라이언트가 네트워크의 상태에 적응하여 지연과 지연변이의 영향을 최소화시키도록 클라이언트의 버퍼관리와 playback control을 설계하였다.

1. 서론

인터넷이 발전함에 따라서 인터넷을 통한 멀티미디어 서비스에 대한 관심도 증가되고 있다. 이로 인해 많은 응용분야들이 나타나게 되었는데, 예를 들자면 VOD (Video On Demand), 비디오 텔레컨퍼런스, 원격교육, 가상현실, 원격진료 등의 시스템들이 있다.[1]

하지만 이러한 서비스를 제공하는데 있어서 현재의 인터넷이 "Best effort" 서비스를 제공한다는 점이 문제가 된다. 또한 사용자의 수가 증가함에 따라 발생하는 네트웍의 혼잡(congestion)문제 때문에, 지연(delay)과, 지연변이(jitter) , 패킷손실(packet loss)등은 실시간 멀티미디어 서비스를 제공하는데 있어 가장 큰 문제점이다.

이러한 문제들을 해결하기 위해 서버에게 알려서 해결하려는 방법은 과중한 부하를 안고 있는 서버의 부하를 가중시키게 된다. 이러한 이유로 본 논문에서는 지연으로 인한 언더플로우(Underflow)/오버플로우(overflow)를 제어하면서 지연에 적응할 수 있는 클라이언트를 설계하고자 한다.

본 논문의 구성은 다음과 같다. 2장에서는 Smooth play와 영상의 상호의존관계를 소개하고 3장에서는 제안하는 클라이언트의 구조를 설명하며, 4장에서는 버퍼의 상태에 따른 메카니즘을 설명하겠다.

*본 연구는 2000년도 두뇌한국21사업 핵심분야 및 과학재단 특정기초 98-0101-06-01-3의 지원을 받았음.

2. 관련연구

2.1. Smooth Play [2,3]

Smooth Play 기법은 클라이언트에서 Network상의 지연과 지연변이로 인해 연속적인 영상의 재현을 지속하기 어려울 때 좀 더 부드러운 움직임을 보여주기 위해서 이미 버퍼링되어 있거나 이후의 프레임을 이용하는 기법을 말한다.

이 방법에는 다음 두 가지 기법이 있다.

방법① 버퍼 레벨에 따른 재현시간을 조절하는 방법

방법② 이미 버퍼링된 프레임을 재현 횟수를 조절하는 방법

위 두 방법에서 얻고자 하는 것은 지연으로 인한 수신측의 언더플로우를 방지하고 더 나은 영상의 연속을 제공하는데 있다. 하지만 이 두 방법의 가장 큰 차이점은 한 프레임의 재생시간이 변동여부이다. 방법①은 버퍼레벨에 따라 프레임의 재생기간을 변동하며 방법②은 프레임의 재현기간은 일정하나 미리 부가적 재현을 하는 기법이다.

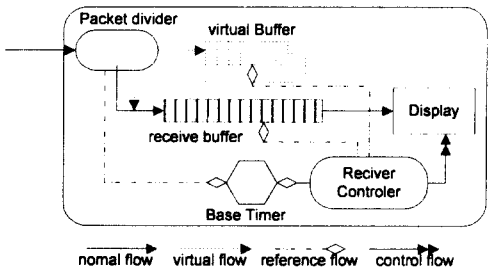
하지만 방법①은 자원 활용도는 높으나 총 재현 시간이 변화하며 버퍼가 안정적인 상태에 있을때도 불필요하게 재현시간이 변화한다는 것이다. 방법②은 시간은 증감은 없으나 재현시간에 늦은 프레임을 버리기 때문에 자원의 활용도가 떨어진다. 따라서 본 논문에서는 두 방법의 장단점을 보안하여 클라이언트를 설계했다.

2.2. 영상의 상호 의존성

전송되는 영상은 MPEG1/2나 H.263과 같은 압축기술을 사용한다. 압축된 영상은 이웃하는 영상을 참조하여 복호화 할 수 있다. 다시 말해서 I ,P, B 타입으로 분류되는데 I-프레임은 독립적이며, P-프레임은 선행하는 하나의 I-프레임이나 P-프레임에 참조해야 한다. 그리고 B-프레임은 선행하는 두 개의 I-프레임이나 P-프레임을 참조해야한다. 만약 P-프레임이 손실되거나 버려진다면, 이 후의 I-프레임은 어떠한 영향도 받지 않지만 이후의 P-프레임이나 B-프레임은 그렇지 않다. 또한 B-프레임이 손실 또는 drop이 되었을 때, I-프레임과 P-프레임, 그리고 이후의 B-프레임 역시 영향을 받지 않는다. 따라서 영상의 연속의 영향을 미치는 정도는 I > P > B이다. 서버 측에서 멀티미디어 스트림을 보낼 때 이러한 프레임간의 상호 의존성을 이용하여 네트워크의 상태가 나빠짐에 따라 상호 의존성이 낮은 타입의 프레임을 점차적으로 버려 네트웍에 적응하는 기법이 제안되어 있다. 본 논문에서는 이 기법을 클라이언트 측에서도 overflow를 방지하기 위해 선택적으로 버리는 방법을 사용한다.

3. 클라이언트의 구조

본 논문에서 제안하고 있는 클라이언트의 구조는 아래의 그림과 같다.

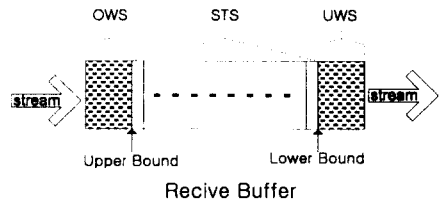


[그림 1] 제시하는 클라이언트의 구조

구성 요소들을 간단히 설명하면, 패킷분류자(Packet divider)는 네트웍으로부터 들어오는 현재의 버퍼의 상태와 패킷의 재현시간을 보고 수용여부를 판단한 후 이에 맞는 버퍼에 할당한다. 수신버퍼(receive buffer)는 실질적인 버퍼이고 가상버퍼(virtual buffer)는 underflow가 발생할 우려가 있는 상태에서 안정된 재현이 가능한 상태로 전이하기 위해 사용되는 일시적이고 가상(추상)적인 버퍼이며 underflow 경계 상태 시에만 수신버퍼에서 재생 기준시간을 기준으로 가상버퍼를 생성되어 사용된다. 수신제어기(Receiver Control) 버퍼들의 상태에 따른 동작을 제어한다.

4. 알고리즘

기존의 방법의 문제점을 보안하기 위하여 본 논문에서 제안하고자 하는 방법은 아래와 같다. 먼저 수신 버퍼의 상태를 크게 세 가지 상태, 즉 안정적 상태(STabile State: STS), underflow 경계 상태(Underflow Warnning State : UWS), overflow 경계 상태(Overflow Warnning State : OWS)로 분류하고 각 상태에 따라 클라이언트의 재현 제어와 버퍼 관리를 달리 한다. 본 논문에서 제시하고 있는 클라이언트의 수신버퍼는 아래의 그림과 같다.



[그림 2] 버퍼의 세 가지 상태

두 임계치를 기준으로 다음과 같은 상태로 구별한다.

ⓐ STS

영상을 기준 시간에 맞게 재생할 수 있는 상태를 말한다. 일반적인 실시간 어플리케이션들과 같이 재현 시간의 늦은 패킷은 버리며 기준 시간에 맞추어 영상을 재현한다.

ⓑ OWS

재생 속도보다 수신하는 속도가 빨라 버퍼가 오버플로우가 발생할 우려가 있는 상태를 말한다.

ⓒ UWS

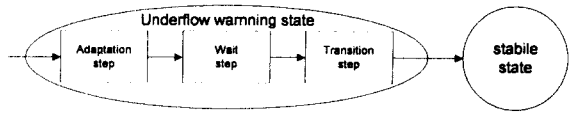
버퍼링 되어 있는 양이 적어 underflow가 발생할 가능성이 있는 상태를 말한다.

이 후에는 UWS와 OWS에서의 동작에 대해서 설명한다.

4.1. UWS에서의 처리

STS에서와는 달리 smooth play를 위해 UWS에 있을 때에는 수신한 어떠한 패킷도 버리지 않는다. UWS에서는 기준을 시간과 버퍼레벨에 두어 한 프레임의 재생기간을 조절한다. 그리고 다른 상태와는 달리 가상(추상)버퍼를 두어, 앞으로 플레이되어야 할 프레임들을 저장한다. 이 가상버퍼는 추가적인 버퍼가 아니라 수신자의 버퍼의 일정 부분을 추상화하고 STS로 전이되었을 때 안정적인 재현을 보장하는 역할을 한다.

UWS에서 STS로의 전이는 다음의 세 단계를 거쳐서 이루어지게 된다.



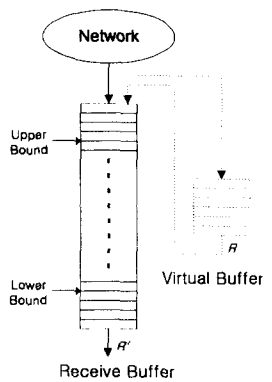
[그림 3] UWS에서 STS로 전이하는 단계

첫 단계는 재현 시간을 수신버퍼와 가상버퍼의 레벨의 합을 현재의 버퍼 레벨로 하여 아래의 수식을 재현 기간을 조절한다. 이 수식은 LB에 수렴하여 underflow를 피하게 된다. 이를 적응 단계(adaptation step)라 한다. 기준 시간에 맞추어 재생할 수 있는 프레임이 발생할 때까지 이 단계는 지속된다.

$$D_i = \alpha R\left(1.5 - \frac{L_{current}}{2LB}\right) + (1-\alpha)\, D_{i-1} \qquad (1)$$

Di : i번째 프레임의 재현시간
α : smoothing factor
R : 초기 재생 속도
LB(Low Bound) : underflow 임계치
$Lcurrent$: 기준이 되는 버퍼 레벨의 현재 값

두 번째 단계는 가상버퍼의 레벨이 충분해질 때까지 수신버퍼의 레벨을 현재의 버퍼 레벨로 하여 식(1)에 따라 동작한다. 그러면 수신버퍼의 레벨은 LB 근접하여 유지된다. 이를 대기 단계라 한다. 아래의 그림은 대기 단계에서의 수신 버퍼와 가상버퍼의 관계를 나타낸 것이다. 실선은 실제의 스트림의 방향이며 점선은 제어를 위한 가상의 스트림의 방향이다. 다시 말하자면 현재 재현이 가능한 프레임은 실제로는 수신버퍼에 저장되지만 개념상 가상버퍼에 저장된다고 생각하게 되는 것이다.



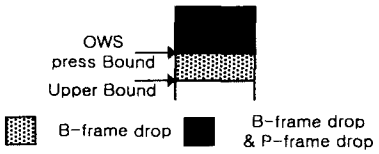
[그림 4] 대기 단계에서의 버퍼

수신버퍼의 레벨은 underflow에 근접하여 있기 때문에 재현속도 역시 초기 재생속도 R에 근접한 값 R'을 갖는다. 그리고 가상버퍼는 기준시간에 의해 제어되기 때문에 R의 속도로 재현시간이 지난 프레임이 수신버퍼로 들어가게 된다. 수신버퍼의 레벨은 거의 일정하다. 따라서 지연변이의 영향은 가상버퍼만 받게되고 가상버퍼의 레벨이 $LB+1$ 이상이 되면 다음 단계로 넘어가게 된다.

세 번째 단계는 가상버퍼 레벨이 충분할 할 때이다. 따라서 수신 버퍼의 있는 프레임들을 빠른 속도로 재생하고 가상버퍼의 첫 프레임은 자신의 재현시간과 현재 진행된 시간과의 차만큼을 추가로 재생한 후 안정적인 상태로 전이한다. 이를 전이 단계라 한다. 이 기준 시간과의 차를 감안해주므로 재생시간에 대한 오차를 감소시킬 수 있다.

4.2. OWS에서의 처리

클라이언트는 OWS에 다른 임계치를 두어 단계적이며 선택적으로 프레임을 버린다.



[그림 5] OWS에서의 선택적 drop정책

먼저 UB(Upper Bound)를 초과하였을 때 상호의존성이 가장 낮은 B-프레임을 가지고 있는 패킷만을 버린다. 그럼에도 불구하고 OWS press Bound를 초과하였을 때는 B-프레임과 P-프레임을 같이 버리므로써 overflow가 발생하지 않도록 하고 영상의 흐름에 영향을 최소화 한다.

5. 결론

본 논문에서는 jitter로 인해 발생할수 있는 버퍼의 상태를 세가지, UWS, STS, OWS,로 나누어 각 상태에 따라 버퍼를 다르게 관리하므로써 네트워크 상태에 잘 적응하는 실시간 멀티미디어 전송에서의 클라이언트를 설계하였다.

UWS에서 사용된 smooth play 기법은 기존의 제안되었던 방법들의 장점을 수용하고 단점을 보안하기 위해 세 단계를 거쳐 처리하게 하였으며 제어를 위해 실제의 수신버퍼의 일부분을 가상버퍼라고 추상화하여 사용하였다.

하지만 본 논문에서 제안하는 클라이언트의 설계는 네트워크의 지연변이만을 고려했기 때문에 그 외의 문제점들도 고려해야할 필요가 있다.

참고문헌

[1] Bobby Vandalore, Wu-chi Feng, Raj Jain, Sonia Fahmy, "A Survey of Application Layer Techniques for Adaptive Streaming of Multimedia," Journal of Real Time Systems (Special Issue on Adaptive Multimedia), January 2000

[2] Dong-Hoon Nam, Seung-Kyu Park, " ADAPTIVE MULTIMEDIA STREAM PRESENTATION IN MOBILE COMPUTING ENVIRONMENT," proceeding of TENCON'99, Cheju, Korea, pp966-969, September 1999

[3] Mourad Daami, Nicolas D., "Client Based Synchronization Control of Coded Data Streams."

[4] Koler D., Muler H., "Multimedia playout synchronization using buffer level control," proceedings of 2nd intl. workshop of IWACA'94, pp.167-180, 1994