True VOD 시스템을 위한 개선된 패칭 알고리즘의 설계

오선진
세명대학교 정보통신학과

Design of an Enhanced Patching Algorithm for True VOD System

Sun-Jin Oh
Dept. of Computer & Information Science, Semyung University

요 약

True VOD 시스템은 클라이언트들이 서비스 요청을 할 때마다 통신망을 통하여 즉시 비디오 서비스를 제공할 수 있는 대화형 시스템이다. 그러나 멀티미디어 스트리밍 응용은 비디오나 오디오 클립들이 높은 대역과 긴 재생시간을 요구하기 때문에 많은 서버 및 네트워크 자원을 요구한다. 패칭기법은 클라이언트들의 요청에 대해 미디어 스트림을 멀티캐스트 전송함으로써 요구되는 서버 및 네트워크 자원을 줄이는 동적 멀티캐스트 방법이다. 하지만 패칭은 클라이언트의 요청을 즉시 처리하기 위해 많은 량의 버퍼링을 필요로 하며 높은 디스크 대역을 요구한다. 본 논문에서는 비디오 서버의 입출력 요청을 동적으로 감소시키며, 인기 있는 비디오에 대한 요청을 즉시 처리할 수 있도록 정규채널과 패칭 채널을 예약해 두는 True VOD 시스템을 위한 개선된 패칭 알고리즘을 제안한다.

1. 서론

주문형 비디오(Video-on-Demand) 시스템은 클라이언트가 안방에서 마음대로 버튼을 눌러서 원하는 미디어를 선택하여 시청할 수 있도록 하는 서비스이다. 따라서 클라이언트가 요구할 때 서비스를 언제든지 통신망을 통하여 제공할 수 있는 대화형 시스템이어야 한다. 이와 같이 클라이언트의 미디어 요청에 대해 즉시 또는 수 초 이내에 전송이 가능하도록 하는 서비스를 True VOD라 한다. 클라이언트들은 비디오 동과 같은 객체를 요청하고, 적절한 지연시간 내에 요청한 객체를 관람하기를 기대한다. 이러한 지연시간은 서비스 요청을 위한 불충분한 대역폭, 디스크로부터 판독 내용을 스케줄링 하기 위한 불충분한 버퍼 공간, 그리고 불충분한 디스크 기억장치 등의 요인으로 발생한다. 지연 요소 중에서 특히 입출력 대역폭은 매우 중요한 자원으로 실시간 처리를 보장하면서 공유를 통하여 기억장치 서버의 입출력 요청을 감소시켜 동시에 서비스할 수 있는 요청 클라이언트의 수를 증가시킬 수 있어야 한다[1, 2]. 주문형 비디오 시스템에서 클라이언트들의 요청은 스트림의 재생기간 동안 특정 인기 비디오로 집중되는 경향을 갖는다[4]. 그리고 비디오 스트림은 엄격한 응답시간이 요구되며, 연속적인 비디오 스트림이 전송되어야 하기 때문에 전송을 위한 입출력 스트림 예약과 채널 할당이 보장되어야 한다. 따라서 경제적이며 대량의 비디오 서비스가 가능하고, 새로운 클라이언트를 위한 처리비용이 최소가 되는 효율적인 스트림 스케줄링 기법이 요구된다.

패칭정책[5, 6, 7]은 새로운 클라이언트 요청을 진행 중인 멀티캐스트와 결합이 가능하게 해 주는 동적 멀티캐스트 방법이다. 요청에 대한 대기시간이 없이 비동기적인 다수의 클라이언트 요청에 대해 비디오 스트림을 멀티캐스트 함으로써, 서버 및 네트워크의 대역을 감소시킨다. 그러나 패칭 정책은 클라이언트의 요청을 즉시 처리하기 위해 높은 디스크 대역 부하와 클라이언트에서의 많은 버퍼링을 요구한다.

본 논문에서는 이러한 패칭정책이 갖는 문제를 해결하기 위하여, 피기백킹 기법을 도입하여 서버의 입·출력 요청을 감소시키고, 인기 비디오에 대한 요청들이 즉시 패치 되도록 패칭채널과 정규채널을 예약해 두는 True VOD 시스템을 위한 개선된 패칭 알고리즘을 제안한다. 본 논문의 구성은 다음과 같다. 2장에서 관련 연구들을 살펴보고, 3장에서 본 논문에서 제안하는 True VOD 서비스를 위한 개선된 패칭 알고리즘을 설명한다. 그리고 마지막으로 4장에서 제안한 알고리즘을 분석하고 결론을 맺는다.

2. 관련 연구

True VOD를 구현하기 위해서는 대용량의 디스크 공간과 빠른 엑세스 시간 그리고 메모리나 시스템 버스 속도도 중요하지만, 네트워크 부하를 감소시키고 여러 전송 채널을 받아들이기에 충분한 입출력과 디스크 대역폭이 필요하다. 여기서는 멀티캐스트를 사용하여 채널을 공유함으로써 네트워크 부하를 줄이는 패칭 방법과 공유를 통하여 저장서버의 입출력 요청을 감소시켜 동시에 서비스되는 클라이언트 요청의 수를 증가시키는 피기백킹 방법을 살펴본다.

2.1 패칭정책

비디오 서버의 가용 채널 수는 매우 제한적이다. True VOD를 서비스하기 위해 중요한 자원인 채널은 많은 클라이언트들에게 공유될 수 있어야 한다. 멀티캐스트를 사용하면 채널을 공유함으로써 네트워크 부하를 줄일 수는 있지만 요청에 대한 대기시간의 증가를 가져온다. 이러한 상반관계를 해결하기 위해 패칭기법이 소개되었다. 하나의 새로운 서비스 요청이 발생되면, 이미 서비스되

고 있는 멀티캐스트 스트림과의 차이(skew point)만큼 새로운 스트림을 전송하고, 그 동안 발생하는 멀티캐스트와의 차이를 클라이언트 버퍼에서 병합하면서 멀티캐스트 스트림을 공유한다. 즉, 패칭은 전체 비디오를 멀티캐스트 하는 것이 아니라 시간차에 의해서 채널의 빠진 부분을 채운다는 의미이다. 패칭하는 방법은 패칭 윈도우를 사용하여 새로운 클라이언트의 요청에 대해 전체 비디오를 전송할 지 여부를 결정한다. 완전한 전송을 시작한 후 패칭 윈도우 시간이상 지난 요청들은 새로운 완전한 전송을 보낸다. 패칭 윈도우 내의 클라이언트 요청들은 가장 가까운 전송을 기준으로 패치된다. 탐욕 패칭 방법은 가능한 한 모든 경우에 패치를 시도한다. 즉, 패칭 윈도우가 비디오의 길이가 된다. 반면에 점진적 패칭은 패칭 윈도우가 클라이언트의 버퍼 공간과 같다. 그러므로 이 경우에는 언제라도 동일한 비디오에 대한 전체 전송이 여러 개 존재할 수 있다. 대체로 점진적 패칭 방법이 탐욕 패칭 방법보다 성능이 우수하다[5, 6, 7].

2.2 피기백킹정책

피기백킹(piggybacking)은 동일한 객체에 대한 입출력 스트림이 하나로 합병될 때까지 진행 중인 스트림의 디스플레이 율을 조정함으로써, 실시간 처리를 보장하면서 공유를 통하여 기억장치 서버의 입출력 요청을 감소시키는 기법이다[3]. 즉, 각 요청에 대해 정상 속도로 디스플레이 하는 것이 아니라, 디스플레이들간의 간격을 좁히기 위해 더 느린 율로 혹은 더 빠른 율로 각 요청의 디스플레이 속도를 조절하게 된다. 이렇게 하여 간격이 없어지면 두 디스플레이 스트림은 하나의 입출력 스트림으로 합병되어 디스플레이 된다. 기억장치 서버에 의해 작동되는 디스플레이 단위는 NTSC (national television system committee) 표준이고, 초당 30 프레임의 율로 디스플레이 된다고 가정한다. 디스플레이 율의 가속과 감속은 비디오에 부가적인 프레임을 추가하고 삭제함으로써 가능하다. 그리고 정상율의 ±5% 범위 내의 디스플레이 율 변경은 클라이언트에 의해 인지되지 않는다. 따라서 실제 디스플레이 율을 5% 낮추거나 높이는 것은 비디오 품질을 손상시키지 않고 가능하다[1]. 그림1은 전체 디

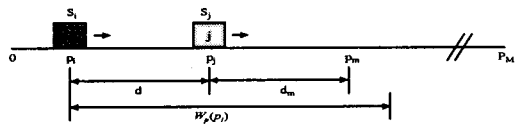

그림 1. 시스템의 디스플레이 상태

스플레이 기간이 P_M인 객체의 디스플레이 상태를 나타낸다. 각 디스플레이 스트림은 그 객체의 디스플레이내의 현재 위치로 식별된다. 즉, 스트림 i는 p_i로 식별되고, 디스플레이 속도 S_i로 이동한다. 입출력 스트림 i와 j를 합병하기 위하여, 먼저 $S_i > S_j$가 보장되어야 하고, 그 다음 합병 기회를 확인하기 위하여 피기백킹이 사용될 수 있는 거리 제한을 정의할 수 있다. 여기서 p_m은 입출력 스트림 i와 j가 합병되는 객체의 디스플레이 위치를, d는 입출력 스트림 i와 j간의 프레임 거리를, d_m은 합병 위치와 j의 현재 위치간의 프레임 거리를 나타낸다. 그리고 격차해소 윈도우(catch-up window), $W_s(p_i)$는 어떤 피기백킹 정책 p에서 스트림 i와 스트림 j를 하나의 스트림으로 합병할 수 있는 최대 거리로 정의되며 다음의 수식 (1)과 같이 계산된다.

$$W_m = \frac{S_{max} - S_{min}}{S_{max}} \cdot L \qquad (1)$$

여기서 L은 비디오의 프레임 길이를 나타내며, $W_p(p_i)$는 객체의 디스플레이에 있어 위치 p_i에 대한 상대적 길이다.

동적 피기백킹은 도착(arrival), 합병(merge), 종료(drop-off), 그리고 윈도우 횡단(window crossing) 등 네 가지 사건 중 하나가 발생할 때 속도를 조정하는 정책들을 고려한다. 합병이 빠를수록 더 많은 자원이 다른 요청들을 서비스하기 위해 사용될 수 있다. 그러므로 정상 속도에서 최대 가능 편차를 가정하여 가장 느린율(S_{min}), 정상율(S_n), 가장 빠른율(S_{max}) 등 세 가지 디스플레이 율을 사용한다. 동적 피기백 합병정책으로는 연속적인 도착 스트림들을 합병하기 위해 짝을 지우는 홀짝 감소정책, 어떤 객체의 디스플레이 시작에서 상대적으로 측정된 최대 합병 윈도우 $W_{sm}^{m}(0)$을 사용하는 단순 병합정책, 그리고 객체의 전체 디스플레이 기간 동안 가능한 많은 입출력 요청을 합병하는 탐욕정책 등이 있다[1].

3. 개선된 패칭 알고리즘

이 절에서는 인기 있는 비디오에 대한 요청들의 실시간 처리를 보장하기 위해 비디오 서버의 입출력 스트림 용량을 예약하고, 패칭정책에서의 높은 디스크 대역폭 부하문제와 많은 버퍼링 요구 등의 문제를 줄일 수 있는 개선된 패칭 알고리즘을 제안한다.

3.1 시스템 모델

본 논문에서는 그림2와 같은 어떤 객체의 각 요청에 대해 하나의 재생 스트림과 이에 대응하는 입출력 스트림이 존재하는 비디오 서버 시스템을 고려한다. 처리 노드들은 디스크들로부터 필요한 데이터를 검색하고, 몇 가지 방법으로 그 데이터의 수정이 가능하며, 재생 스트림들을 사용하여 네트워크를 통해 적절한 상영장치로 전송하기 위한 입출력 스트림을 사용한다. 저장서버의 입출력 요구는 같은 객체에 대한 요청들에 대응하는 다수의 재생 스트림들을 서비스하기 위해 하나의 입출력 스트림을 사용함으로써 감소될 수 있다. 그리고 저장서버는 요청의 디스플레이 율을 동적으로 변경할 수 있는 기능, 즉, 어떤 객체의 디스플레이의 일부분을 동적인 시간 압축 또는 시간 확장의 기능을 갖는다고 가정한다.

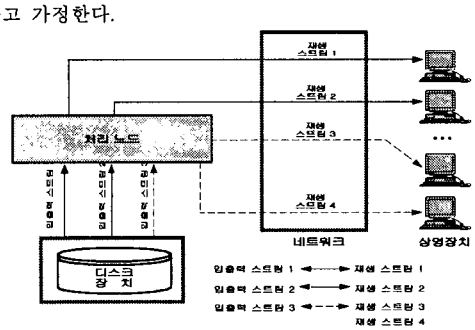

그림 2. 비디오 서버 시스템 모델

3.2 개선된 패칭 알고리즘의 시나리오

제안한 패칭정책은 동적 멀티캐스트를 수행하는 패칭정책과 자원 예약기반 피기백킹 정책의 장점을 모두 수용한 정책이다. 개선된 패칭정책에서 통신채널은 전체 비디오를 멀티캐스트하는 정규 멀티캐스트와 비디오의 스큐 부분만 멀티캐스트하는 패칭 멀티캐스트가 사용된다. 전자의 경우는 정규채널에 의해 전송되고, 후자의 경우는 패칭채널에 의해 전송된다. 클라이언트는 각 채널로부터 정규 스트림 또는 패칭 스트림을 다운로드 받는다. 패칭을 위해 클라이언트는 각 채널에 대한 스트림을 다운로드 하는 두 개의 로더 L_R과 L_P를 가지며, 버퍼에 저장된 비디오 프레임을 조정하여 스크린에 상영시켜 주는 *Video Player*가 필요하다. 그림3은 패칭 정책의 시스템 개요도를 보여준다. 이 그림은 클라이언트 A, B, C가 순차적으로 동일한 비디오에 대한 요청을 하였고, 모두 패칭 윈도우 내에 위치하는 경우이다. 클라이언트 A는 정규 스트림만 전송 받으며, 클라이언트 B는 현재 패칭 스트림의 전송이 끝나고 정규 스트림의 전송

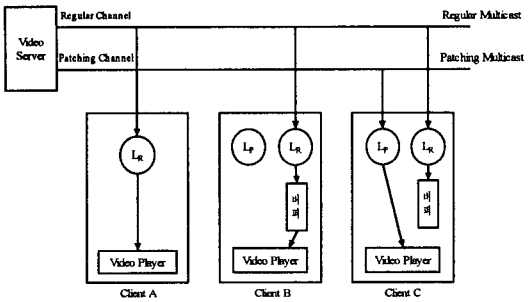

그림 3. 패칭정책의 시스템 개요도

을 받고 있으며, 클라이언트 C는 가장 최근에 도착한 요청으로 패칭 스트림의 재생을 하고 있는 상태를 보여 준다[5].

개선된 패칭 알고리즘의 시나리오는 다음과 같다. 우선 클라이언트는 요청토큰 *(ClientID, VideoID)*을 송신하여 비디오를 요청한다. 여기서 *ClientID*는 클라이언트 자신의 주소이고, *VideoID*는 요청된 비디오의 ID이다. 비디오 서버는 요청한 비디오가 인기 비디오이면 예약채널을 할당하고, 그렇지 않으면 비예약채널을 할당한다. 여기서 비디오 i의 인기도는 Zipf의 법칙[8]을 따른다고 가정한다. 비디오 관람 패턴에 대한 통계에 의하면 비디오에 대한 인기도는 특정 비디오에 편중되는 것으로 나타난다. 이런 지역성(locality)은 Zipf의 법칙을 사용하여 표현할 수 있다. Zipf의 법칙에 의하면 M 비디오 중에서 n번째로 인기 있는 비디오를 선택할 확률은 C/n이고, 이때 C = 1/(1+ 1/2 +1/3 +·····+1/M)이다.

비디오 서버가 서비스 할 준비가 되면 스케줄링 할 비디오와 동일한 비디오 스트림이 느린 속도(S_{min})로 격차해소 윈도우 내에 재생되고 있는지 확인한다. 여기서 최대 격차해소 윈도우는 수식 (1)에 의해 구할 수 있고, x 단위 시간만큼 떨어진 스트림들이 합병되는 시점은 $t_m = xS_{min}/(S_{max} - S_{min})$으로 계산된다. 만약 격차해소 윈도우 내에 비디오 스트림이 존재하면 정규채널을 통해 그 스트림과의 합병을 위하여 빠른 속도(S_{max})로 재생을 시작한다. 존재하지 않으면 패칭 윈도우 내에 같은 비디오 스트림이 재생되고 있는지 확인한다. 여기서 패칭 윈도우는 하나의 비디오에 대한 정규 멀티캐스트를 시작한 시간부터 패칭이 적용되는 시간까지를 말한다. 만약 패칭 윈도우 내에 동일한 비디오 스트림이 재생되고 있으면 패칭채널을 통해 재생을 시작한다. 하지만 존재하지 않으면 정규채널을 통해 재생을 시작한다. 이 때 비디오의 재생은 인기 비디오의 경우 예약채널을 통해 이루어진다. 비디오의 재생을 시작하기 전에 비디오 서버는 서비스 토큰 *(PID, RID)*을 클라이언트에게 통보한다. 여기서 PID는 패칭채널의 ID를, 그리고 RID는 정규채널의 ID이다. 클라이언트가 서비스 토큰을 수신하여 검사 후의 시나리오는 다음과 같다.

1. PID가 NULL이면, 서버는 채널 RID에 정규 멀티캐스트를 시작한다. 클라이언트는 비디오 데이터를 수신 받기 위해 다운 로더 L_R만 활성화한다. 그리고 전송된 데이터는 *VideoPlayer*에 의해 비디오 프레임들이 스크린에 보여진다.

2. PID가 NULL이 아니면, 서버는 채널 RID에 패칭 멀티캐스트를 실행한다. 클라이언트는 RID와 PID 모두에게서 데이터를 동시에 다운로드 받기 위해서 다운 로더 L_R과 L_P 모두를 활성화한다. *VideoPlayer*는 먼저 PID로부터 도착하는 패칭 스트림을 재생하면서 RID로부터 도착하는 정규 스트림은 일시적으로 국부 버퍼에 캐시한다. 패칭 멀티캐스트의 전송이 끝나면 *VideoPlayer*는 비디오 파일의 나머지 부분을 다운 로더 L_R로 계속 다운로드 하여 국부 버퍼에서 교환하여 캐시 함으로써 연속 재생을 가능하게 한다.

개선된 패칭 정책의 시나리오는 그림4와 같다. 여기서 *a, b, c, d, e*는 요청 비디오에 대한 입출력 스트림을, 그리고 $W_m(0)$는 격차해소 윈도우를 각각 나타낸다. 그림에서 지금 막 도착한 요청 비디오 b는 격차해소 윈도우 내에 느린 속도로 선행되고 있는 동일한 비디오 스트림이 존재하므로 정규채널을 통해 빠른 속도로 재생을 시작하여 진행되고 있는 미디어 스트림과의 합병을 시도한다. 그러나 선행되고 있는 동일한 비디오 스트림이 격차해소 윈도우 내에 존재하지 않으면 패칭 윈도우 내에 진행되고 있는 동일한 비디오 스트림이 존재하는지 확인한다. 만약 동일한 비디오 스트림이 패칭 윈도우 내에서 진행되고 있으면 패칭채널을 통해 재생을 시작하고, 선행되고 있는 비디오 스트림을 클라이언트에서 버퍼링 한다. 스큐부분에 대한 재생이 완료되면 패칭채널을 통한 재생을 중단하고, 국부버퍼에 저장된 비디오 스트림으로 연속해서 재생을 진행한다. 패칭 윈도우 내에 같은 비디오 스트림이 존재하지 않을 경우에는 정규채널을 통해 새로운 비디오 요청에 대한 서비스를 시작한다. 인기 있는 비디오 스트림 요청들은 패칭 윈도우 내에 존재할 확률이 높으므로 그 스트림들을 패칭하기 위하여 서버 채널용량을 예약해 둔다. 그림에서 b와 c는 인기 있는 비디오 스트림이고, a, d, e는 인기 없는 비디오 스트림을 나타낸다.

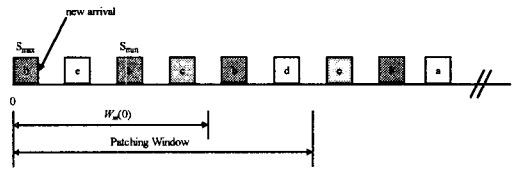

그림 4. 개선된 패칭 알고리즘의 시나리오

채널 예약용량은 패칭채널 예약용량과 정규채널 예약용량으로 이루어진다. 각 채널 예약용량은 수식 (2)와 (3)으로부터 구할 수 있다.

- 패칭채널 예약용량 : $\sum_{i=1}^{k} p_i \times PW \times \lambda$ (2)
- 정규채널 예약용량 : $k \times (\frac{L}{PW} - 1)$ (3)

여기서 p_i는 비디오 i가 요청될 확률, L은 비디오의 길이, PW는 패칭 윈도우 크기, λ는 도착율, 그리고 k는 서버 채널용량이 예약되는 인기 있는 비디오의 개수를 나타낸다. 이 논문에서 제안하는 개선된 패칭 정책의 패칭 윈도우 알고리즘과 서버 알고리즘은 각각 그림5와 그림6과 같다.

Algorithm *WindowPatching(FreeID, RegularID)*
Begin
If $t-t_s \le PW$
the workload for channel *FreeID* is V[$t-t_s$];
Else
FreeID is designated to start a new regular multicast as follows;
Modify service token as*(PID=NULL, RID=NULL)*;
Set the workload for new regular channel *FreeID* as V[|V|].
End
where,
t : current time.
t_s : start time of the regular multicast on channel RegularID.
V : the video currently multicast on channel RegularID.
|V| : playback duration of the video V.
PW : size of patching window.

그림 5. 패칭 윈도우 알고리즘

```
Algorithm Server Main Routine
      in Reservation-based Piggybacking Patching Algorithm.
Begin
   reserve the I/O channel capacity of video server for the
   hottest videos requests;
   select the next video, say v, to serve according to FCFS
   policy;
   If a video v is hot video
     dispatch a free channel from reservation channel;
   Else
     dispatch a free channel from non-reservation channel;
   initialize the service token as (PID=NULL, RID=NULL);
   If there is no regular multicast of video v in progress,
     set RID=FreeChannel;
   Else If a stream j within W is moving at Smin
          Si = Smax;
          RID=FreeChannel;
   Else
     Si = Smin;
     set PID=FreeChannel and RID=LatestRegular;
    // LatestRegular is the latest regular channel for video v.
     call WindowPatching(FreeChannel, LatestChannel);
    // to determine the portion of video data which should be
       multicast on FreeChannel.
   For each request token (ClientID=vClient, VideoID=v) in
       WQ.
     If PID=NULL
       append vClient to the client list of FreeChannel;
     Else
       append vClient to the client list both FreeChannel and
              LatestRegular;
       send Service token to vClient;
       delete the request token from WQ;
   activate the Multicast on FreeChannel;
  Case merge of stream i and j :
        drop stream i ;
        Sj = Smin ;
End.
```

그림 6. 개선된 패칭 알고리즘

4. 분석 및 결론

주문형 비디오 시스템은 다수의 클라이언트가 서비스를 요청하면 제한된 자원을 가지고 연속 매체들의 실시간 처리를 보장하면서 언제든지 통신망을 통해 서비스를 제공할 수 있는 대화형 시스템이어야 한다. 클라이언트의 미디어 요청에 대해 즉시 혹은 수 초 이내에 전송이 가능한 주문형 비디오 서비스를 True VOD라 한다. True VOD를 구현하기 위해서는 무엇보다도 네트워크 부하를 감소시키고 여러 전송채널을 받아들이기에 충분한 입출력과 디스크 대역폭이 필요하다. 디스크 입출력 대역폭은 서비스의 지연시간을 결정하는 중요한 자원이다. 멀티캐스트는 서버의 대역폭 사용을 감소시키는데 우수한 기법이다. 하지만 대기시간을 증가시키는 단점 때문에 True VOD 서비스를 제공하지 못한다. 패칭정책은 이러한 대기시간과 멀티캐스트의 상반관계를 해결하기 위해 제안되었다. 패칭은 전체 비디오를 멀티캐스트하는 것이 아니라 미디어의 스큐 부분만큼만 패칭채널을 이용하여 채워줌으로서, 네트워크 부하를 줄일 수 있고 대기시간 없이 서비스가 가능하게 되므로 True VOD 서비스의 구현이 가능하게 하였다. 그러나 패칭은 별도의 패칭채널을 사용함으로써 최대 두 배의 채널용량을 요구하며, 클라이언트에서의 많은 버퍼공간을 필요로 하는 문제점을 가지고 있다. 비디오 서버의 한정된 채널용량으로 인하여, 인기 있는 비디오의 요청이 상대적으로 많기 때문에, 패칭될 수 있는 기회가 많지만, 채널용량의 부족으로 패칭 기회를 잃어버리는 경우가 빈번하게 발생할 수 있다.

본 논문에서는 대기시간과 멀티캐스트의 상반관계를 해결하기 위해 패칭 방법을 기반으로, 인기 있는 비디오에 대한 클라이언트의 요청이 자주 발생되고, 또한 이들 요청들이 패칭될 수 있는 기회가 많다는 특징을 감안하여, 인기 있는 비디오의 요청을 원활하게 서비스하기 위한 비디오 서버 채널을 예약해 두는 개선된 패칭 방법을 제안하였다. 이렇게 함으로써 인기 있는 비디오에 대해 채널용량의 부족으로 인한 패칭 기회를 잃어버리는 경우를 줄일 수 있다. 패칭정책의 또 다른 문제점인 별도의 패칭채널을 사용함으로써 많은 량의 채널용량을 요구하는 문제는 비슷한 시기에 도착하는 동일한 비디오에 대한 클라이언트의 요청들을 동일한 객체에 대한 입출력 스트림이 하나로 합병될 때까지 진행중인 비디오 스트림의 디스플레이율을 조정하는 피기백킹 정책을 이용하여 크게 줄일 수 있다. 일반적으로 인기 있는 비디오들에 대한 클라이언트들의 요청은 비슷한 시기에 집중되기 쉬우므로 피기백킹 정책을 도입함으로써 여러 클라이언트들의 요청들을 하나의 멀티캐스트로 합병할 수 있어, 요구되는 채널용량을 크게 줄일 수 있다. 이때 조정되는 비디오 스트림의 디스플레이 율은 최대 ±5% 범위 내에서 이루어지므로 재생되는 비디오 스트림의 서비스 품질에는 영향을 주지 않는다. 그리고 피기백킹 정책을 도입함으로써 요청 비디오 스트림에 대한 패칭의 기회가 줄어들고, 채널용량이 절감되며, 따라서 클라이언트에서의 비디오 스트림에 대한 버퍼링이 필요 없게 되어 국부 버퍼의 사용을 크게 완화할 수 있다. 결과적으로, 본 논문에서 제안하는 개선된 패칭 알고리즘은 True VOD를 구현하면서, 인기 있는 비디오에 대한 채널용량을 예약함으로써 인기 있는 비디오 요청에 대한 원활한 패칭을 보장하였고, 피기백킹 정책을 도입함으로써 동일한 비디오 객체에 대한 입출력 스트림을 합병하여 요구되는 채널용량을 크게 줄일 수 있고, 클라이언트에서의 버퍼링을 완화함을 알 수 있다. 향후 연구과제는 본 논문에서 제안하는 개선된 패칭 알고리즘에 대한 모의 실험을 통한 성능평가를 하고, 비디오 서버의 부하에 상관없이 적절한 수준의 이탈율을 유지하는 동적 일괄처리에 관한 것이다.

참 고 문 헌

[1] L. Golubchik, J. Lui, and R. Munts, "Reducing I/O Demand in Video-On-Demand Storage Servers," *In Proceedings of the ACM Sigmetrics '95*, pp.25-36, 1995.

[2] A. Dan, D. M. Dias, R. Mukherjee, and D. Sitaram, "Buffering and Caching in Large-Scale Video Servers," IBM Research Division, Technical Report RC 19903, 1995.

[3] C. Arrarwal, J. Wolf, and P. S. Yu, "On Optimal Piggyback Merging Polices for Video-On-Demand Systems," IBM Research Division, Technical Report RC 20337, 1996.

[4] K. A. Hua and S. Sheu, "Skyscraper Broadcasting : A New Broadcasting Scheme for Metropolitan Video-on-Demand Systems," *In Proceedings of the ACM SIGCOMM'97*, Cannes, France, 1997.

[5] K .A. Hua, Y. Cai, S. Sheu, "Patching : A Multicast Techniques for true Video-On-Demand Services," *ACM Multimedia '98*, 1998.

[6] S. Sen, L. Gao, J. Rexford, and D. Towsley, "Optimal Patching Schemes for Efficient Multimedia Streaming," Technical Report 99-2, Dept. of Computer Science, University of Massachusetts Amherst, 1999.

[7] Y. Cai, K. Hua, and K. Vu, "Optimizing Patching Performance," *In Proceedings of the SPIE/ACM Conference on Multimedia Computing and Networking*, 1999.

[8] G. K. Zipf, *Human Behavior and the Principles of Least Effort*, Addison-Wesley, 1949.

[9] 오선진, 이경숙, 배인한, "비디오 서버를 위한 예약기반 하이브리드 디스크 대역폭 절감정책의 설계 및 평가", 한국정보처리학회 논문지 B 제 8-B권 제 5호, pp. 523 - 532, 2001.