VOD 시스템에서의 Interchange Agent 운영 알고리즘

강석훈* · 박수현**

Interchange Algorithm for VoD System

Seokhoon Kang* · Suhyun Park**

요 약

기존 VoD 시스템의 멀티캐스트 방법은 사용자의 요청수가 증가함에 따라 시스템 부하가 증가하게 되어 시스템의 성능이 떨어지게 된다. 본 논문에서는 사용자들이 빈번하게 요청하는 인기비디오에 대한 정보를 프록시 서버에 저장 관리하여, 사용자들의 비디오 요청 시 발생하는 네트워크의 사용과 서버의 부하를 줄이는 방법에 대해 연구하였다. 본 논문에서는 효율적인 VoD 서비스를 위해 멀티 프록시(Multi-Proxies)들을 관리하는 Interchange Agent(IA)를 구축하였다. 이를 위해, 멀티캐스팅 기법을 이용하여 사용자의 동일 비디오 요청을 하나로 묶어 처리하여 네트워크 및 서버의 부하를 줄고, 자신의 리스트를 이용하여 멀티프록시 캐시의 내용을 중복 없이 관리하여 시스템 효율을 높이며, 스위칭 기능을 통하여 사용자가 요청한 비디오가 연결된 다른 프록시에 존재하면 이를 스위칭 채널을 통해 즉시 전송하여 실시간 서비스의 효율을 높이고, 각 프록시 캐시의 임시저장장소와 카운터를 이용하여 사용자의 인기비디오 순위 변경 시, 캐시의 내용을 변경하여 그 흐름을 반영하도록 하였다.

ABSTRACT

This paper proposes a approach to configure efficient video-on-demand system by introducing Multicast and Cache Video-on-Demand (MCVoD) system. As a key element of the MCVoD system, ***interchange agent*** provides this system with multicasting and switching functions. With the multicasting, the MCVoD system is able to reduce the load on the network as well as VoD servers by transmitting only one video request instead of sending multiple requests on a same video stream. The switching enables clients to receive the first stream of requested video streams instantly without waiting time and also allows avoiding undesirable duplication of video streams in the system. With various experiment results through simulation about waiting time and cache hit ratio, we show that the MCVoD system employing the interchange agent provides better performance than current uni-proxy based system.

키워드

VOD 시스템, Interchange Agent, 멀티 프록시, 멀티 캐스팅

Ⅰ. 서 론

VoD 시스템 분야의 연구는 VoD 서버의 채널 할당, 에이전트를 이용한 멀티프록시의 관리 및 멀티프록시의 사용 등 여러 분야가 있다[1-9].

[11]에서는 VoD 서버의 채널 할당방법에 관해 다루

* 인천대학교 멀티미디어시스템공학과

** 동서대학교 컴퓨터정보공학부 컴퓨터공학전공

접수일자 : 2005. 9. 20

고 있다. 채널 할당방법은 Batching, Patching, Batched Patching의 세 가지 방법으로, Optimal cutoff threshold를 이용하여 VoD 서버의 bandwidth를 서비스 요청 도착율 (λ)에 관계없이 $O(\sqrt{L})$로 제한됨을 보이고 있다. 또한, 실시간 비디오 서비스를 제공하기 위해 단일프록시 캐시 및 클라이언트의 디스크 공간을 활용하는 방법을 제안하고 있다. 이 공간에 Prefix라는 인기비디오의 첫 번째 또 몇몇 스트림을 저장하여 두어, 인기비디오가 요청되면 즉시 제공하여 실시간 VoD 서비스를 제공한다. 그러나 단일 프록시를 사용할 경우 사용자의 인기비디오 우선순위를 변경하면 이를 적응적으로 반영할 수 없으며, 따라서 실시간 서비스를 제공하지 못하는 결과를 가져올 수 있다는 단점을 가지고 있다.

[10]에서는 스위칭 에이전트를 이용한 Head-End Node의 관리를 보여준다. Head-End Node는 스위칭 에이전트의 관리 하에 인기비디오의 모든 스트림을 저장하여, 서버 측의 비디오 서비스 요청 도착율을 줄이는 데 목적이 있다. 스위칭 에이전트 및 다중 Head-End Node를 통해 인기 비디오 요청하는 경우, 요청에 대한 서버의 채널 할당 없이 Head-End Node를 통해 비디오 스트림을 제공함으로써 전체 네트워크의 부하를 줄이는 결과를 보여준다.

스위칭 에이전트를 이용하여 다중 Head-End Node의 중복 없이 관리하고, 사용자의 인기 비디오 순위 변경 시에도 이를 적응적으로 반영하여 실시간 서비스를 구현한다. 그러나 인기비디오의 전체 스트림을 모두 다중 Head-End Node에 저장함으로 시스템 구성비용이 증가할 수 있다.

시스템 구성비용은 다중 Head-End Node에 인기 비디오 전체 스트림을 저장하기 위한 공간 비용과 비디오의 인기 순위 변경 시, 이를 반영하기 위해 비디오 스트림 교체 할 때 발생한다. 따라서 서버의 채널할당에 대한 네트워크 부하를 줄이는 장점을 가져올 수 있는 동시에 전체 시스템 비용을 높이는 결과를 가져올 수 있다.

본 논문에서 제안한 MCVoD 시스템에서는 효율적인 VoD 서비스를 위해 멀티 프록시들을 관리하는 IA를 구축하였다. IA는 다음과 같은 4가지의 특징을 가진다. 첫째, 멀티캐스팅 기법을 이용하여 사용자의 동일 비디오 요청을 하나로 묶어 처리하여 네트워크 및 서버의 부하를 줄인다. 둘째, 자신의 리스트를 이용하여 멀티프록시 캐시의 내용을 중복 없이 관리하여 시스템 효율을 높인다. 셋째, 스위칭 기능을 통하여 사용자가 요청한 비디오가 연결된 다른 프록시에 존재하면 이를 스위칭 채널을 통해 즉시 전송하여 실시간 서비스의 효율을 높인다. 넷째, 각 프록시 캐시의 임시저장 장소와 카운터를 이용하여 사용자의 인기비디오 순위 변경 시, 캐시의 내용을 변경하여 그 흐름을 반영하도록 한다.

본 논문에서는 시뮬레이터를 구축하여 MCVoD 시스템의 성능을 평가하였다. 멀티캐스팅 기법 및 캐시 미스로 인해 발생하는 지연시간 문제를 해결하기 위한 방법의 성능을 평가하고, IA와 단일프록시의 전체 채널 및 사용자의 비디오 요청 비율에 따른 지연시간을 비교하여 성능을 분석하였다. 또한 프록시의 효율성을 측정하기 위해, 사용자의 비디오 요청 비율에 따른 프록시 Hit 수를 비교하였다.

Ⅱ. Interchange Agent의 사용 채널

IA는 사용자의 요청을 처리하는 멀티캐스팅 기능과 다중 프록시를 연결하기 위한 스위칭 기능을 수행한다. 멀티캐스팅은 멀티캐스트 윈도우(MW)들의 동일한 요청을 묶어 하나의 채널(ch)로 처리한다. 스위칭 기능은 클라이언트의 요청이 연결되어 있지 않은 프록시의 캐시 상에 존재할 경우, 해당 스트림을 IA를 경유하여 전송하는 기능이다. 즉, 첫 번째 스트림을 받기 위한 새로운 단일캐스트 채널 하나와 클라이언트로 첫 번째 스트림을 전송하기 위한 새로운 단일 캐스트 채널 하나를 할당하게 된다.

따라서 IA의 평균 채널은 멀티캐스트를 위한 평균 채널과 스위칭을 위한 평균 채널의 합으로 나타낼 수 있다.

LA_B= 멀티캐스트을 위한 평균 채널 + 스위칭을 위한 평균 채널

그림 1은 IA의 평균 채널 할당량을 나타내고 있다.

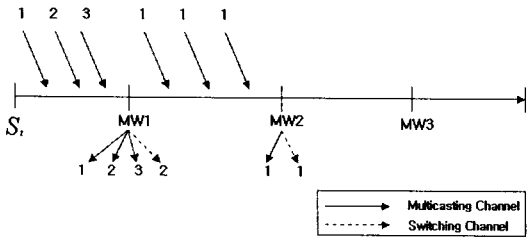

그림 1. interchange Agent의 효율성
Fig. 1 The performance of Interchange Agent

그림 1에서, MW는 IA의 멀티캐스트 윈도우이고, 비디오 1번, 2번은 인기비디오, 비디오 3번은 비인기 비디오라고 가정한다. IA는 비디오 서비스 요청을 처리하기 위해 첫 번째 멀티캐스트 윈도우를 시작하고 MW1 동안 인기비디오 요청 2개와 비인기비디오 요청이 1개가 도착한다고 가정한다. 또한, 두 번째 비디오 2번이 요청되었을 때, 이미 비디오 스트림이 다른 프록시에 존재한다고 가정한다.

이 경우에 IA는 서로 다른 비디오 요청(비디오 1, 2, 3)을 처리하기 위해 멀티캐스트 채널을 3개 할당하고 인기 비디오 2번을 위해 스위칭 채널을 할당한다. 스위칭 채널은 프록시로부터 비디오 2번의 스트림을 받기 위해 채널 1개와 비디오 서비스를 요청한 사용자에게 비디오 스트림을 전송하기 위해 채널 1개를 할당한다.

MW2 동안 3개의 동일한 인기 비디오 서비스 요청 중에서 2번은 비디오 1번의 첫 번째 스트림을 저장하고 있는 프록시에 연결된 사용자의 서비스 요청이고 나머지 하나는 다른 프록시에 연결된 사용자의 서비스 요청이라고 가정한다. 이 경우 멀티캐스팅을 위한 채널은 1개 필요하고 스위칭을 위한 채널은 2개 필요하다.

전체 비디오 서비스 요청의 도착율을 λ라고 할 때 K개의 인기 비디오를 위한 요청 도착율은 $\lambda_k = \lambda \times (80/100)$이며, N-K개의 비인기 비디오 도착율은 $\lambda_{N-K} = \lambda \times (20/100)$이 된다. 이때 80/100은 Zipf 분포에 따른 것이다.

따라서 요청 도착율에 따른 새로운 단일캐스트 채널의 수는 수식 (1)이 된다.

$$\sum_{n=1}^{\lambda_k} ch + \sum_{n=1}^{\lambda_{N-K}} (\Theta R_{n-1} \neq R_n then ch = 1) \quad (1)$$

R_n은 K개의 인기 비디오 중 사용자가 요청한 하나의 비디오이며, 동일한 비디오의 요청은 하나로 처리하게 된다.

스위칭을 위한 채널의 수는 새로운 단일캐스트 채널 2개와 다른 프록시 상에 존재하는 인기 비디오의 요청 도착율을 곱한 $2 \times \lambda_{K-10}$이 된다. MW1 상의 멀티캐스트와 스위칭을 위한 채널의 수는 수식(2)로 나타낼 수 있다.

$$[\sum_{n=1}^{\lambda_k} ch + \sum_{n=1}^{\lambda_{N-K}} ch] + (2 \times \lambda_{K-10})(\Theta R_{n-1} \neq R_n then ch = 1) \quad (2)$$

따라서 비디오 길이 L분 동안 멀티캐스트 윈도우 m개로 나누었을 때 IA의 전체 평균 채널은 수식(3)로 나타낸다.

$$IA_B = \frac{\sum_{m=1}^{m}((\sum_{n=1}^{\lambda_k} ch + \sum_{n=1}^{\lambda_{N-k}} ch) + (2 \times \lambda_{K-10})}{m} (\Theta R_{n-1} \neq R_n then ch = 1) \quad (3)$$

예를 들어 그림 1의 IA의 채널을 계산하여 보면 다음과 같다. MW1 동안 인기 비디오를 위해 할당한 멀티캐스트 채널은 2개이며 비인기 비디오를 위해 할당한 멀티캐스트 채널은 1개이다. 이때 2번 비디오를 위해 스위칭 채널을 한개 할당하였다. MW2 동안 인기 비디오를 위해 할당한 멀티캐스트 채널의 수는 1개이며, 스위칭 채널을 위해 할당한 채널의 수는 1개이다. 따라서 두개의 멀티캐스트 윈도우 동안의 IA의 평균 채널은 4가 된다.

Ⅲ. Interchange Agent 운영

IA의 운영절차는 크게 멀티캐스팅 부분 및 다중 프록시의 관리 부분으로 나누어 설계한다.

3.1 멀티캐스팅

클라이언트의 비디오 서비스 요청은 다중 프록시를 거쳐 IA로 전달된다. IA는 멀티캐스트 윈도우동안 비디오 서비스 요청을 모은 후, 동일 비디오 서비스 요청을 하나로 묶어 멀티캐스트 라우터로 전달한다. 그림 2는 멀티캐스팅 과정을 나타낸다.

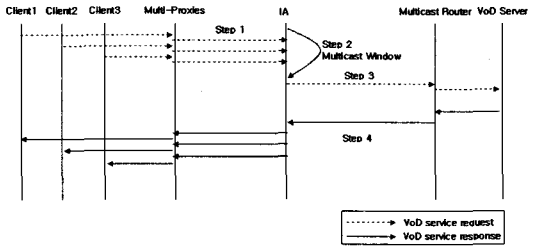

그림 2. 멀티캐스팅
Fig. 2 Multicasting

3.2 캐시 관리

비디오 서비스 요청이 전달되면 IA는 요청된 비디오의 첫 번째 스트림을 저장하고 있는 프록시를 찾고 클라이언트에게 서비스를 시작함으로 시간 지연 없는 실시간 서비스를 시작한다. 멀티프록시의 캐시는 비디오의 인기 순위에 기반 한 클라이언트의 요청에 맞게 그 내용이 가변적으로 변경되어야 한다.

각 프록시의 캐시는 n(n=10)개의 비디오 스트림을 저장할 수 있으며, 1개의 임시저장장소(temporary buffer)를 가지고 있다. 또한 각 비디오 스트림에 대한 서비스 횟수를 저장하기 위한 카운터가 있고, 이 를 이용하여 캐시의 내용을 변경 할 수 있다.

멀티프록시는 캐시 초기화(Cache Initiation), 캐시 Hit, 캐시 Miss, 캐시 대치(Cache Replacement)의 4가지 동작을 바탕으로 캐시를 관리하게 된다.

3.2.1 캐시 초기화

캐시 초기화는 멀티프록시의 모든 캐시가 비어있을 경우(예를 들어 초기상태)에 대한 처리를 포함한다. 그림 3은 캐시 초기화 과정이고 방법의 상세는 다음과 같다.

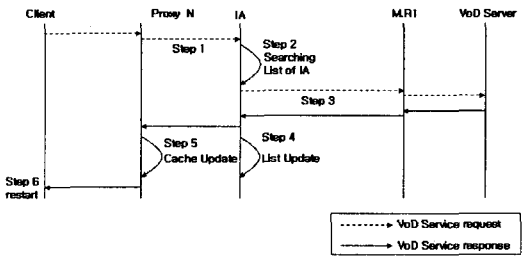

그림 3. 캐시 초기화
Fig. 3 Cache Initiation

<방법 1>

단계1 클라이언트의 비디오 서비스 요청을 프록시 N을 거쳐 IA에 전송한다.

단계2 IA는 리스트에서 요청받은 비디오 첫 번째 스트림을 검색한다.

단계3 동일 비디오 요청은 하나로 묶어 멀티캐스트 라우터를 거쳐 VoD 서버로 전송한다.

단계4 VoD 서버는 비디오 스트림을 IA에 전송하고 IA는 프록시 N에 전송하고 리스트에 요청한 비디오 목록을 저장한다.

단계5 프록시 N은 첫 번째 스트림을 캐시에 저장하고 카운터를 1 증가시킨 후, 클라이언트에게 비디오 스트림을 전송한다.

단계6 위의 과정을 반복 수행한다. 만약 프록시 N의 캐시가 다 찬 상태에서 클라이언트가 다른 비디오 스트림을 요청하면, VoD 서버로부터 비디오 스트림을 전송받아 비어있는 프록시 (프록시 M)의 캐시에 첫 번째 스트림을 저장하고 카운터를 증가시킨다. IA는 각 프록시의 캐시의 내용을 중복 없이 저장하도록 관리한다.

3.2.2 캐시 Hit

클라이언트가 요청한 비디오가 IA의 프록시에 있는 경우이고, 그림 4는 이 경우에 비디오 요청을 서비스하는 과정을 나타내었다.

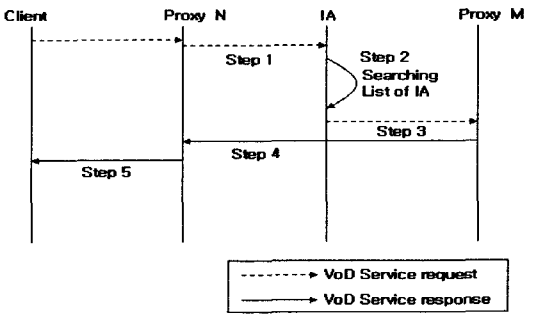

그림 4. 캐시 Hit
Fig. 4 Cache Hit

서비스 처리 방법은 <방법 2>이다. 단계1과 단계2는 클라이언트가 서비스를 요청하는 과정으로 <방법 1>과 같다.

<방법2>

단계3 해당 비디오 스트림이 프록시 M에 있으면, 프록시 M에 비디오 스트림을 프록시 N으로 전송하도록 명령한다.

단계4 비디오 스트림을 프록시 N에 전송한다.

단계5 프록시 N은 클라이언트에게 해당 비디오 스트림을 전송한다.

3.2.3 캐시 Miss

요청된 비디오가 프록시에 존재하지 않을 경우이고 처리과정은 그림 5이고 상세 방법은 <방법 3>과 같다.

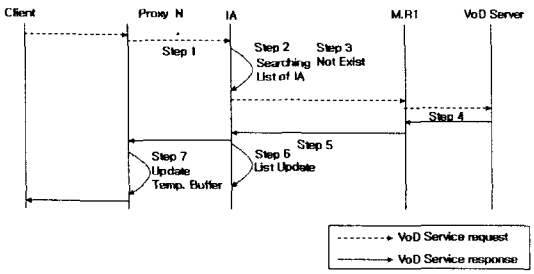

그림 5. 캐시 Miss
Fig. 5 Cache Miss

<방법 3>

단계3 프록시에 요청한 비디오 스트림이 없으면 캐시 미스가 발생한다.

단계4 IA는 비디오 서비스 요청을 멀티캐스트 라우터를 거쳐 VoD 서버로 전송한다.

단계5 VoD 서버는 해당 비디오 스트림을 멀티캐스트 라우터를 거쳐 IA로 전송한다.

단계6 IA는 프록시 N에게 해당 비디오 스트림을 전송하고 요청한 비디오 목록을 리스트에 저장한다.

단계7 프록시 N은 임시저장장소에 첫 번째 비디오 스트림을 저장하고 해당 카운터를 1증가 시킨 후, 클라이언트에 비디오 서비스 전송한다. 만약 프록시 N의 임시저장장소에 다른 비디오 스트림이 있으면 삭제하고 전송 받은 비디오의 첫 번째 스트림을 저장하고 카운터를 1 증가시킨다.

3.2.4 캐시 교체

관리되지 않은 사용자의 새로운 요청이 발생할 때 캐시의 내용을 업데이트 하는 과정이다. 그림 6에 처리과정을 나타내었고, 방법의 상세는 다음과 같다.

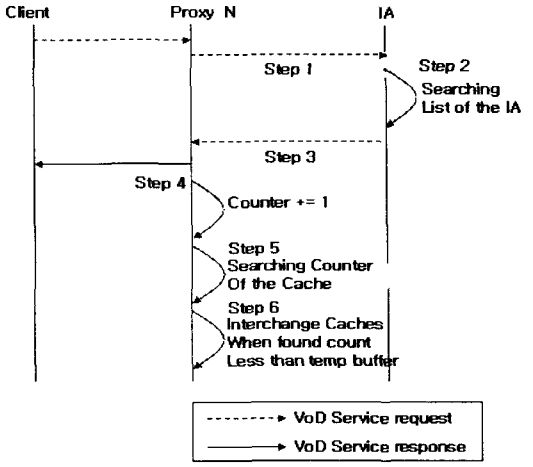

그림 6. 캐시 교체
Fig. 6 Cache Replacement

<방법 4>

단계3 프록시 N에 요청한 비디오 서비스의 첫 번째 스트림이 있으면, 프록시 N에게 클라이언트에게 비디오 스트림을 전송하도록 한다.

단계4 프록시 N은 캐시의 임시 저장 장소의 스트림을 클라이언트에게 전송하고 해당 카운터를 1 증가 시킨다.

단계5 프록시 N은 캐시의 카운터를 검색하여 임시저장장소의 카운터보다 적은 값을 가진 저 장장소가 있는지를 검색한다.

단계6 위와 같은 작업을 반복적으로 수행하여 프록시의 캐시 중 임시저장장소의 카운터보다 적은 값을 가진 저장장소가 있으면 해당 저장장소의 스트림과 임시저장장소의 스트림을 교체하는 작업을 수행한다.

Ⅳ. 시뮬레이션 및 성능평가

본 논문에서는 MCVoD 시스템의 IA와 단일프록시의 비교를 통해, 시뮬레이션 결과를 분석하였다. 시뮬레이션 환경은 다음과 같다.

IA의 멀티프록시는 각 10개의 인기비디오 첫 번째

스트림을 저장하고 있고 각 1개의 임시 저장공간을 가지고 있다. 즉, 총 33개의 캐시들로 구성되어 있다. 단일프록시의 경우는 IA와 동일한 구성을 위해 총 33개의 인기비디오의 첫 번째 스트림을 저장하도록 구성하였다.

인기비디오의 첫 번째 스트림의 길이는 1분으로 설정 하였으며, 각 비디오의 요청 비율은 도착율 λ를 가지는 포아송 분포(Poisson Distribution)을 따르도록 하였다. 사용자의 비디오 선택은 Zipf 분포를 따르도록 하였다.

표 1. 시뮬레이션 환경
Table1. Simulation Environment

인수	값	범위
Number of Videos	100개	변화없음
Prefix Length	1분	변화없음
Request Rate	50 /분	분당 50에서 100번
Number of Channels	50개	50에서 100개
IA Multicast Window	1분	변화없음
Uni-Proxy Window	1분	변화없음
Skew Factor	0.271	변화없음

따라서 각 비디오의 요청 비율을 다음과 같이 설명할 수 있다. Zipf 분포에 따라 i번째 인기비디오의 요청 가능성(Probability of choosing)을 ρ_i라고 할 때, 요청 가능성은 수식 4에 나타내었다.

$$\rho_i = \frac{f_i}{\sum_{j=1}^{N} f_i} \quad (4)$$

$f_i = 1/i^{1-\theta}$,

$i = 1,...N$ (N; 전체 비디오의 개수)

Θ; skew factor

본 실험에서는 $\theta = 0.271$으로 설정하였다[11]. 따라서 i번째 인기비디오의 요청 비율은 수식 5와 같이 나타낼 수 있다.

$$\lambda_i = \rho_i^{\lambda} \quad (5)$$

사용자의 요청 비율이 1분을 기준으로 이루어지며 프록시 캐시에 저장된 인기비디오의 첫 번째 스트림의 길이를 1분으로 설정하였다. IA의 멀티캐스트 윈도우는 1분으로 설정 하고 단일프록시의 요청 처리 시간도 1분으로 설정하였다. 총 시뮬레이션 시간은 IA가 제공하는 첫 번째 비디오 스트림의 길이에 따라 시뮬레이션 시간은 100분으로 하였다.

4.1 비디오 요청 비율에 따른 지연시간

MCVoD 시스템은 멀티캐스팅 기법을 통해 발생하는 지연 시간 문제를 해결하기 위하여, 멀티프록시를 제공하고 있으며, 멀티프록시의 관리를 위해 IA를 이용하고 있다. 본 실험에서는 IA를 이용하여 멀티프록시와 단일프록시의 사용자 요청 처리 지연 시간을 비교해 봄으로써, 실시간 서비스 제공에 따른 성능평가를 수행하였다.

표 2. 요청율과 지연시간
Table2. Request Rate vs. Waiting Time

Request rate (Perminute)	50	60	70	80	90	100
IA Waiting time	58.72	58.92	59.16	59.4	59.58	59.82
Uni-Proxy Waiting time	76.92	77.34	77.76	78.12	78.42	78.6

위의 시뮬레이션은 전체 채널의 수를 50으로 고정하고 사용자의 비디오 요청비율을 50부터 100까지 증가하도록 하였다.

표 2를 통해, IA의 지연시간은 1분 미만이며, 단일프록시의 경우 1분 이상임을 알 수 있다. IA의 멀티캐스트 윈도우 및 단일프록시의 사용자 비디오서비스 요청처리시간을 1분으로 설정하였으므로 실시간성을 보장함을 알 수 있다. 이러한 결과는 MCVoD 시스템의 멀티캐스팅 방식으로 인해 발생할 수 있는 지연시간 문제를 IA를 통한 멀티프록시의 관리를 통해 해결할 수 있음을 보여 준다.

4.2 채널의 양에 따른 지연시간

IA는 멀티캐스팅 및 스위칭 기능을 제공하고 있다. 따라서 멀티캐스팅 기능을 통해 사용자의 동일 비디오

요청의 경우 하나의 요청으로 묶어 전송함으로써 IA의 전체 채널의 양을 감소시키고 있다.

또한 스위칭 기능을 통해 단일프록시의 채널에 비해 IA에 부가적인 채널이 추가된다. 그러나 스위칭 기능을 통한 사용자의 요청한 비디오가 멀티프록시상의 사용자와 연결되지 않은 다른 프록시에 존재 할 때, 이를 전송하므로 지연시간 문제를 해결하고 있다.

본 실험은 사용자의 서비스 요청 비율을 50으로 고정하고, IA 및 단일프록시의 전체 채널의 양을 50부터 100까지 증가시키도록 하였다.

표 3. 채널수와 지연시간
Table3. Number of Channels vs. Waiting Time

Number of Channels	50	60	70	80	90	100
IA Waiting time	58.74	43.38	29.22	18.72	9.24	0
Uni-Proxy Waiting time	76.92	61.2	55.2	49.38	45.96	41.94

위의 결과를 통해 IA의 경우에 총 채널의 양이 100개가 넘어가면 사용자의 지연시간은 없음을 알 수 있고, 스위칭 기능을 통한 부가적인 채널의 추가가 필요하지만 멀티프록시의 인기비디오순위 반영을 통해 지연시간이 줄어듦을 알 수 있다. 단일프록시의 경우 채널의 양이 60이 넘어가면서 서비스 지연시간이 서비스 처리시간 1분보다 적게 나타나 실시간 서비스를 할 수 있음을 알 수 있다. 하지만 채널의 양을 100개까지 증가함에도 불구하고 지연시간이 0이 되지는 않음을 볼 수 있다.

4.3 프록시 Hit수

MCVoD 시스템은 IA를 이용하여 카운터를 기반으로 한 임시 저장공간을 관리하고 있다. 임시 저장공간에 있는 첫 번째 비디오 스트림의 요청에 따라 해당 프록시의 임시 저장공간의 카운터와 나머지 10개의 저장공간의 카운터를 비교한다. 이 요청이 증가할수록 임시 저장공간의 카운터가 나머지 10개의 저장공간의 카운터 보다 큰 경우에 해당 비디오의 첫 번째 스트림을 교체하는 기능을 수행한다.

IA는 멀티프록시의 사용자 인기비디오 순위 반영 및 스위칭 기능을 이용하여 멀티프록시의 Hit 수를 증가시킨다.

표 4. 요청율과 Hit 수
Table4. Request Rate vs. Number of Hits

Request rate	50	60	70	80	90	100
IA Hits	4918	4934	4970	4992	5016	5029
Uni-Proxy Hits	2178	2183	2175	2178	2183	2172

위의 결과를 통해 IA의 프록시 Hit 수는 요청율에 관계없이 단일프록시의 Hit 수에 비해 두배 이상 많음을 알 수 있다. 또한 사용자의 요청 비율이 증가함에 따라 IA가 관리하는 멀티 프록시의 Hit 수가 증가하지만, 단일프록시의 Hit수는 크게 차이가 없음을 알 수 있다.

결과적으로 프록시의 Hit수를 통하여 사용자의 인기 비디오 순위의 변경 시 이를 반영하고 있음을 시뮬레이션으로 보인다. 단일프록시가 아닌 멀티프록시를 사용함으로써 사용자의 비디오 요청을 실시간으로 서비스할 수 있다.

V. 결 론

MCVoD 시스템의 IA는 멀티캐스팅과 스위칭 기능을 제공하며 멀티프록시를 중복 없이 관리 하는 역할을 담당한다. 멀티캐스팅 기능을 통해 사용자의 동일 비디오 요청을 하나로 묶어 전송함으로써 네트워크의 부하를 줄인다.

또한 멀티캐스팅으로 인해 발생하는 지연시간문제를 해결하기 위해 IA 관리하의 멀티프록시를 사용한다. 멀티프록시는 인기비디오의 스트림 중 각 10개의 첫 번째 스트림 저장하고 있으며 카운터를 기반으로 한 임시저장장소 하나를 가지고 있다. 따라서 하나의 프록시는 총 11개의 첫 번째 스트림을 저장할 수 있으며, 하나의 IA는 총 3개의 멀티프록시를 관리하고 있다.

사용자가 인기비디오를 요청할 때는 멀티프록시에 저장되어 있는 인기비디오의 첫 번째 스트림을 즉시 전송해줌으로써 실시간 VoD 서비스를 제공할 수 있도록 한다. IA는 카운터를 기반으로 한 임시저장장소의 관리를 통해 사용자의 인기비디오 순위 변경을 통한 임시저장장소의 카운터 증가 시 해당 프록시의 나머지 카운터와 비교하여 캐시의 내용을 변경하는 작업을 수

행한다. 따라서 사용자의 인기비디오 순위를 멀티프록시내에 반영함으로써 사용자의 인기비디오 요청 시 지연시간문제를 최소화하고 있다.

IA는 멀티프록시의 효율적인 사용을 하기 위해, 스위칭기능을 제공한다. 이러한 스위칭기능을 통해, 사용자의 비디오 요청이 다른 프록시에 존재할 때, 첫 번째 스트림을 스위칭을 통해 즉시 제공해줌으로써 사용자의 지연시간문제를 해결하고 있다. 이러한 스위칭 기능을 바탕으로 IA는 멀티프록시의 내용을 중복 없이 유지할 수 있도록 하고 있다.

일반적으로 VoD 시스템은 멀티캐스팅을 통해 발생하는 지연시간문제를 해결하기 위해 프록시의 캐시를 활용하고 있다. 단일프록시의 사용을 통해 지연시간문제를 해결하는 방법을 사용하고 있지만 이러한 단일프록시를 여러 개 묶어 관리함으로써 그 효율을 높일 수 처리한다. 동시에 인기비디오 순위를 멀티프록시상에 반영함으로써 사용자의 비디오 요청 시 첫 번째 스트림을 즉각적으로 제공함으로써 프록시 캐시 미적중의 발생에 따른 지연시간문제까지 해결함을 보였다.

본 논문은 MCVoD 시스템상의 IA의 성능을 분석 및 운영절차를 바탕으로 시뮬레이션을 통해 단일프록시와 멀티프록시를 관리하는 IA를 비교하여, IA를 사용한 멀티프록시가 동일한 크기의 단일프록시보다 우수한 성능을 보임을 나타내었다.

참고문헌

[1] Huadong Ma, Kang G. Shin "Multicast Video-on-Demand services" January 2002 ACM SIGCOMM Computer Communication Review, Volume 32 Issue 1, p31~43

[2] T.D.C. Little, D. Venkatesh, "Prospects for interactive video-on-demand", IEEE Multimedia, 1(3), 1994, pp.14-24

[3] Lee J.Y.B, Lee C.H "Design, performance analysis, and implementation of a super-scalar video-on-demand system", IEEE trans. on circuits and systems for video technology, vol. 12, no. 11, Nov. 2002, p983~p997

[4] Lin Huang Chang, Kuen-Chu Lai "Near video-on-demand systems with combining multicasting and unicasting batching", IEEEE, Electrical and Electronic Technology, 2001.

[5] Choi, C.Y, Hamdi, M. "A scalable video-on-demand system using multi-batch buffering", IEEE, Global Telecommunications Conference, 2001.

[6] Chen-Lung Chan, Te-Chou Su, Shih-Yu Huang and Jia-Shung Wang "Cooperative proxy scheme for large-scale VoD systems", IEEE, Parallel and Distributed Systems, 2002.

[7] L. gao, D. Towsley. "Supplying instantaneous video-on-demand services using controlled multicast." In Proc. of IEEE International Conference on Multimedia Computing and Systems, Florence, Italy, Jun 1999.

[8] K. A. Hua, Y. Cai, and S. Sheu. Patching: A multicast technique for true video-on-demand services. In Proc. of ACM SIGCOMM, Sept 1997.

[9] James F. Kurose "Computer Networking : A Top-Down Approach Featuring the Internet." ublished by Addison-Wesley

[10] SH Kang, YS Woo, BH Kim and IS Kim, "VOD Service using Web-Caching Tech on the Head-End-Network", LNCS2668, Springer Verlag

[11] Ramesh, S. Injong Rhee and Guo. K. "Multicast with cache (Mcache): an adaptive zero-delay video-on-demand service" , IEEE, INFOCOM 2001.

저자소개

강석훈(Seok-Hoon Kang)

한양대학교 전자통신공학과 공학박사(1995)
동서대학교 컴퓨터공학과 조교수 (1996~2003)
인천대학교 멀티미디어시스템공학과 조교수(2004~현재)
※관심분야 :임베디드 시스템, 차세대이동통신

박수현(Suhyun Park)

부산대학교 전자계학과 이학박사(1999)
동서대학교 컴퓨터공학과 조교수 (1996~현재)
Utah State University 방문교수 (2003. 1~2004. 8)
※관심분야 : 임베디드 시스템, e-Learning 시스템, 유비쿼터스 헬스케어