Head-End-Network에서 Web-Caching을 사용한 VOD 서버☆

VOD Server using Web-Caching in Head-End-Network

김 백 현* 황 태 준** 김 익 수***
Backhyun Kim Taejune Hwang Iksoo Kim

요 약

본 논문은 멀티미디어 서버의 과도한 부하는 물론 네트워크 자원의 비효율적인 사용과 같은 문제점을 해결하기 위한 단말-네트워크(HNET) 내에 새로운 분산 웹-캐싱 전략에 대한 다루고 있다. 제안된 분산 웹-캐싱은 인근에 위치하는 단말-노드들에 분산 망을 구현하여 특정 단말-노드에 부하가 집중되지 않고, HNET의 분산 단말-노드들에 복사본이 존재하지 않는 장점을 갖고 있다. 제안된 웹-캐싱 기법은 각 단말 노드들이 동일한 비디오 서비스를 요청할 경우 HNET 내의 스위칭 에이전트의 제어아래 서버로부터 비디오의 일부 스트림만을 분산 저장하며, 서비스의 제공은 클라이언트들이 분산 단말-노드에 교호적으로 접속하여 서비스 받도록 한다. 새로운 Web-caching 기법에 사용된 제거 알고리듬은 LFU, LRU와 단말간 지연을 줄이기 위해 비디오의 첫 번째 스트림을 마지막으로 제거하는 방식을 조합하여 시뮬레이션을 수행하였다.

Abstract

This paper presents distributive web-caching technique on the Head-End-Network (HNET) to solve excessive load of multimedia server and inefficient use of network resources. The proposed web-caching is not concentration of load for a specific Head-End-Node(HEN), and it has advantage that there is no copy of a specific item on distributive HENs.

This technique distributively stores on HENs partial streams of requested videos from clients connected to HENs and the order of store streams follows the order of request of identical video items from HENs. Thus, storing streams on each HEN that requests service are different. When a client requests the cached video on some HENs, the HEN that connects him receives streams in the order of store from HENs which stored them and it transmits them to him. These procedures are performed under the control of SA. This technique uses cache replacement algorithm that the combination of LFU, LRU and the last remove for the first stream of videos in order to reduce end-to-end delay.

Keyword : Web caching, VOD, Streaming

1. 서 론

인터넷상에서 텍스트는 물론 오디오와 비디오를 포함하는 멀티미디어 데이터를 주문형(on-demand)으로 서비스하는데 있어서 가장 문제가 되는 것은 멀티미디어 서버의 과도한 부하와 네트워크 자원의 비효율적인 사용이다. 현재 인터넷에서 음악과 비디오 데이터에 대한 스트리밍 서비스 요청이 폭주하고 있으며, 주문형 서비스를 원활하게 제공하기 위해서 서버는 강력한 처리능력을 보유하고 있어야만 한다[1]. 또한 네트워크는 대량의 멀티미디어 데이터를 전송하기 위해 고속의 광대역 망이 필수적으로 구성되어야만 하는 어려운 문제점들을 갖고 있으며, 이를 해결하기 위해서는 투자가 이루어져야 하기 때문에 시간이 필요하고, 이를 위한 다양한 연구가 진행되고 있다[2,3].

* 정 회 원 : 인천대학교 정보통신공학과 박사과정
hidesky24@incheon.ac.kr(제 1저자)
** 정 회 원 : 인천대학교 정보통신공학과 박사수료
tjhwang@incheon.ac.kr(공동저자)
*** 정 회 원 : 인천대학교 정보통신공학과 교수
iskim@incheon.ac.kr(공동저자)
☆ 본 연구는 한국과학재단 지정 인천대학교 멀티미디어 연구센터의 지원에 의한 것입니다.

현재 이들 문제점을 해결하기 위한 가장 활동적인 연구로서 멀티캐스트 전송을 사용한 서비스와 클라이언트들 근처에 프락시(proxy)를 사용한 웹-캐싱 전략을 사용하여 멀티미디어 서비스 제공하려 한다.

첫째, 멀티캐스트 전송을 사용한 멀티미디어 서비스는 클라이언트들의 동일 아이템에 대한 서비스 요청 시점이 비슷할 경우 이들을 그룹화하여 서비스하는 방법으로서 클라이언트들의 서비스 요청을 그룹화하기 때문에 멀티미디어 서버의 부하를 줄일 수 있으며, 네트워크 자원을 보다 효율적으로 사용할 수 있지만 서비스 요청 후 일정 시간을 대기하여야 할뿐만 아니라 아직 인터넷상의 많은 라우터들이 멀티캐스트를 지원하지 않고 있어 인프라가 부족한 실정이다[4,5,6].

둘째, 프락시를 사용한 웹-캐싱 전략은 클라이언트로부터 한번 요청된 데이터를 프락시에 저장하여 이후의 동일한 서비스 요청은 프락시에 이미 저장되어 있는 데이터로 서비스를 제공할 수 있어 원격의 멀티미디어 서버를 통하지 않기 때문에 서버의 부하를 줄일 수 있을 뿐만 아니라 서비스 요청 후 서비스가 제공되기까지의 단말간 지연(end-to-end delay)을 최소화 할 수 있다[7,8]. 그러나 기존의 웹-캐싱 전략은 인기가 있는 아이템을 저장하고 있는 프락시들은 상대적으로 트래픽이 폭주하는 단점을 내포하고 있으며, 또한 인근의 프락시에 동일한 아이템들의 복사본이 다수 존재한다는 단점을 갖고 있다[9,10].

본 논문에서 제안하는 새로운 웹-캐싱 전략은 현재 우리나라에 광범위하게 도입되어 있는 ADSL과 케이블 모뎀 망을 이용하여 이를 아파트 단지별로 네트워크 화하는 단말-네트워크(Head-End-Network : HNET)를 도입하여 프락시 역할을 수행하는 단말-노드(Head-End-Node : HEN)를 두어 보다 원활한 멀티미디어 서비스를 제공하게 된다. 따라서 HNET은 다수의 단말-노드들과 이에 직접 연결되는 클라이언트 그룹과 HNET을 제어 감독하는 스위칭 에이전트(switching agent : SA)로 구성된다. 다수의 단말-노드들은 소량의 버퍼를 두며, 단말-노드에 접속되어 있는 클라이언트들의 모든 서비스를 관할하도록 한다.

또한 제안한 웹-캐싱 전략은 특정 아이템에 대해 클라이언트가 서비스를 요청할 경우 접속되어 있는 단말-노드는 자신의 버퍼에 일정양의 비디오 스트림들을 저장해야 하기 때문에 인기가 있는 아이템들이 특정 단말-노드에만 저장되는 일이 없기 때문에 특정 단말-노드에 부하가 집중되지 않는 즉 부하가 균등하게 배분될 뿐만 아니라 이들 단말-노드들 사이에 동일한 아이템이 중복되어 저장되지 않는 특징을 보유하고 있다.

본 논문의 2장에서는 분산 단말-노드들을 사용한 HNET의 구조와 동작에 대해 다루고, 3장에서는 단말-노드의 캐싱 전략-버퍼의 용량 제한에 따른 캐시-미스, 캐시-히트 및 캐시 제거전략에 대해서, 4장에서는 시뮬레이션과 결과의 분석을 그리고 5장에서는 결론의 순서로 기술된다.

2. Head-End-Network의 구조와 동작

인터넷상에서 대량의 멀티미디어 데이터에 대한 주문형 서비스를 제공하는데 있어서 과도한 서버의 부하를 감소시키고, 부족한 네트워크 자원을 보다 효율적으로 사용할 수 있는 웹-캐싱을 수행하기 위한 네트워크로서 우리는 HNET을 구성하며, 이의 구조는 스위칭 에이전트와 다수의 단말-노드와 소량의 버퍼 그리고 단말-노드에 연결되어 있는 다수의 클라이언트들로 이루어지며 이를 그림 1에 나타내었다.

영화와 같은 멀티미디어 데이터를 인터넷과 HNET로서 전송하는 VOD 서버는 클라이언트들로부터 서비스 요청을 스위칭-에이전트와 단말-노드들을 통해 받아들이며, 채널의 여유가 있는 한 요청된 비디오 영화에 대해 즉각적으로 서비스를 제공하기 위해 스트리밍 서비스를 수행하는 소스 디바이스이다.

HNET은 스위칭-에이전트에 연결되어 있는 서로 인접해 있는 단말-노드(HEN 또는 LAN)들을

서로 연결한 망이라 할 수 있다. 따라서 본 논문에서 단말-노드 망이란 대학이나 연구소 등의 다수의 LAN 망을 연결한 네트워크나 몇 개 아파트 단지의 ADSL망 또는 케이블 모뎀 망을 연결한 네트워크로 설명될 수 있을 것이다.

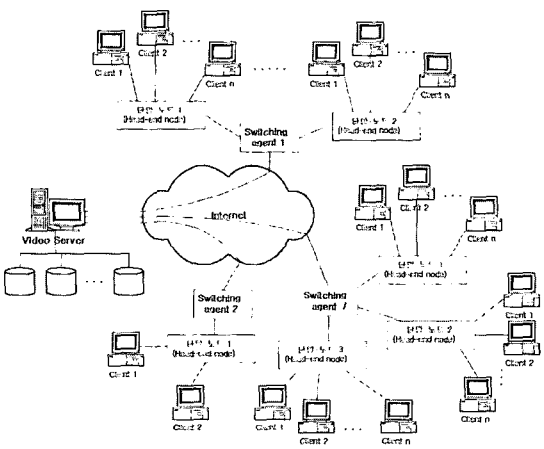
HNET 내의 스위칭-에이전트는 각 단말-노드에 연결되어 있는 클라이언트들의 서비스 요청에 따라 현재 단말-노드상의 버퍼에 요청된 비디오의 스트림들이 이미 저장되어 있는가의 여부를 판단하여 VOD 서버로의 연결을 유지할 것인가의 여부를 결정한다. 또한 SA는 클라이언트들의 요청에 따라 VOD 서버로부터 전송되는 요청 비디오 스트림들이 서비스를 요청한 단말-노드들에 분산 저장되도록 제어한다. 이밖에 SA는 클라이언트들이 인근의 단말-노드들에 이미 저장되어 있는 비디오들에 대해서 서비스를 요청할 경우 이들 단말-노드들로의 스위칭을 제어한다.

클라이언트들이 직접 연결되어 있는 단말-노드들은 클라이언트들의 서비스 요청에 대해 스위칭-에이전트를 통해서 VOD 서버에 접속함과 동시에 자신의 버퍼에 요청된 비디오의 첫 번째 스트림이 저장되어 있는지의 여부를 판단하여 이미 저장되어 있을 경우 서비스를 즉각적으로 개시하지만, 그렇지 않을 경우 스위칭-에이전트에 제어를 요청하게 된다.

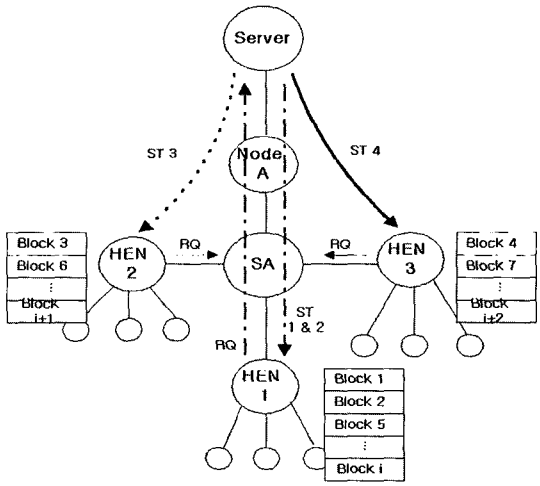
그림 1에 나타낸 HNET 내의 클라이언트들의 서비스 요청에 따른 단말-노드들의 버퍼에 요청된 비디오 스트림들이 캐시 되는 동작을 그림 2에 나타냈으며, 그림 3에 단말-노드들에 요청된 몇 개 비디오 스트림들의 저장결과를 나타냈다.

그림 2는 단말-노드 1(HEN1), 2와 3에 접속되어 있는 클라이언트들이 특정 비디오에 대해 서비스를 요청할 경우 스위칭-에이전트는 자신의 목록을 조사한 후 인근의 단말-노드에 비디오 스트림이 저장되어 있지 않을 경우 VOD 서버로부터 해당 스트림들이 각 단말-노드들에 저장되는 방법을 나타내고 있다. 그림 2에서 단말-노드 1은 서버로부터 데이터 스트림 1을 전송받아 버퍼에 저장함과 동시에 서비스가 이루어질 수 있음을 알 수 있다. 스트림 1이 단말-노드 1의 버퍼에 저장되는 동안 다른 단말-노드들로부터 동일 비디오에 대한 서비스 요청이 없지만 두 번째 스트림이 서버로부터 전송되는 동안에 단말-노드 2와 3으로부터 차례대로 서비스 요청이 있을 경우 데이터 스트림 2는 단말-노드 1에 전송되지만 데이터 스트림 3과 4는 각각 단말-노드 2와 3에 저장되고, 이후의 스트림들은 이들 3개 단말-노드에 순차적으로 저장되게 된다.

따라서 단말-노드 1에 연결된 클라이언트가 서비스



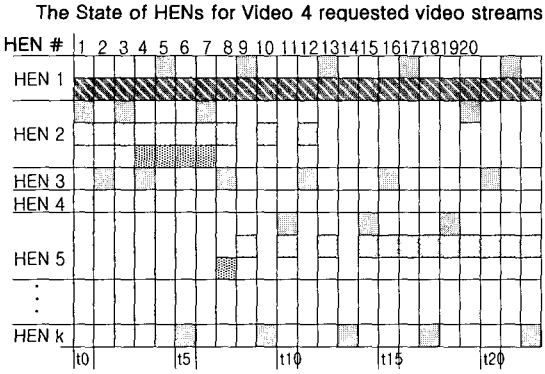
(그림 1) Head-End-Network의 기본 구조



(그림 2) 단말-노드의 캐싱 전략

를 요청할 경우 서버로부터 전송되는 비디오 스트림(예: 스트림 1, 2)은 일단 단말-노드 1의 버퍼에 순차적으로 저장됨과 함께 서비스를 요청한 클라이언트에 즉시 스트리밍 서비스가 이루어진다. 이후 다른 단말-노드 2(HEN 2)로부터 동일한 비디오에 대해 서비스를 요청할 경우에는 서버로부터 전송되는 이어지는 비디오 스트림 3은 단말-노드 2에 저장된다. 따라서 단말-노드 1의 클라이언트는 스트림 3을 서비스 받기 위해 스위칭-에이전트의 제어 하에 단말-노드 2에 접속하여 서비스를 받으며, 이와 동시에 단말-노드 2의 클라이언트는 스트림 1과 2를 서비스 받기 위해 단말-노드 1에 접속하여 서비스를 개시하고 이어서 자신의 버퍼로부터 나머지 스트림을 서비스 받는 방식을 사용한다.

그림 3은 그림 2와 같은 방법으로 단말-노드 2와 3의 클라이언트가 비디오 i(▩)를 각각 t=t0과 t0<t<t1 사이에, t2<t<t4 에서 단말-노드 1과 k가 동일 비디오를, 단말-노드 5가 t6<t<t10에 서비스를 요청하고, 그리고 t6<t<t10에 단말-노드 5와 2가 서비스 요청을 철회하며 t<t19에 단말-노드 2가 다시 요청을 할 경우의 각각의 단말-노드상에 비디오 i(▩)의 일부 스트림들이 각각의 버퍼에 저장되어 있는 그림이다. 이와 같은 새로운 웹-캐싱 기법은 VOD 서비스에 있어서 인기가 있는 비디오의 경우 보다 많은 클라이언트들이 서비스를



(그림 3) 각 단말-노드에 저장되어 있는 서비스 요청된 비디오 스트림

요청하기 때문에 인기 비디오에 대해 일정 분량의 데이터 스트림들이 각기 다른 단말-노드에 저장되어 자동적으로 서버의 부하는 물론 단말-노드들의 부하 분산이 이루어질 수 있는 장점을 보유하고 있다.

3. 단말-노드의 캐시 관리

클라이언트가 VOD 서버에 특정 비디오에 대해 서비스를 요청할 경우 단말-노드와 스위칭 에이전트를 거쳐 일단 서버에 연결을 요청하게 되며, 이와 동시에 단말-노드와 스위칭-에이전트는 각각 자신의 목록을 조사하여 요청된 비디오 스트림들이 인근의 단말-노드들에 이미 캐시되어 있는지를 여부를 확인한다. 이 확인작업을 통해 이미 캐시되어 있을 경우(캐시-히트) 스위칭-에이전트는 서버로의 접속을 끊고 해당 단말-노드로의 스위칭을 제공한다. 그러나 단말-노드 어디에도 사전에 저장되어 있지 않는 경우(캐시-실패)와 초기상태의 경우 서버로의 연결은 유지되며, 서버로부터 전송되는 요청 비디오에 대한 스트림들은 서비스를 요청한 단말-노드로 서비스 요청 순서에 따라 분산 전송함과 동시에 스위칭-에이전트의 저장목록에 이 결과를 등록한다. 그림 3은 바로 이를 도식적으로 나타낸 결과이다.

그림 4에 캐시-실패에 따른 처리 알고리듬을 나타냈다.

단말-노드들에 연결된 클라이언트들의 특정 비디오에 대한 서비스 요청에 대해 단말-노드상의 버퍼에 요청된 스트림들이 이미 저장되어 있을 경우인 캐시-히트의 경우 분산 단말-노드들은 마치 서버와 같이 동작하며 2장에서 설명한 바와 같이 스위칭-에이전트의 제어아래 서비스는 즉각적으로 개시될 수 있다. 이 경우 비록 비디오의 첫 번째 부분이 다른 단말-노드에 저장되어 있을 경우라도 이들 분산 단말-노드들은 동일 공간 내에 위치할 수 있기 때문에 지연은 최소화된다. 캐시-히트에 대한 처리절차는 그림 5와 같다.

Step 1: 클라이언트는 단말-노드와 SA를 통해 VOD 서버에 서비스 요청
Step 2: SA는 VOD 서버와의 연결 유지
Step 3: 서비스를 요청한 단말-노드와 SA는 각각 비디오의 첫 번째와 이어지는 스트림들이 자신의 버퍼에, 그리고 다른 단말-노드에의 저장 여부를 조사
Step 4: 캐시-실패의 경우 SA는 연결 유지
캐시-히트의 경우 캐시-히트 처리절차로 이동
Step 5: SA는 VOD 서버로부터 첫번째 비디오 스트림 및 이어지는 스트림을 대기
SA는 다른 단말-노드들로부터 동일한 비디오에 대한 서비스 요청유무를 판단
Step 6: SA는 첫 번째 스트림을 서비스를 요청한 단말-노드로 전송
단일 단말-노드 요청일 경우 이어지는 스트림을 계속 저장토록 제어
다중 단말-노드 요청일 경우 이어지는 스트림들을 단말-노드의 서비스 요청 순서에 따라 분산저장
SA와 단말-노드들은 저장 결과를 각각의 목록에 등록
Step 7: 클라이언트들은 SA의 제어아래 스트림들이 저장되어 있는 단말-노드들에 접속
step 8: 단말-노드는 다른 단말-노드들로부터 전송되는 스트림들을 저장하여 등록서비스를 개시

(그림 4) 캐시-실패 알고리듬

Step 1: 클라이언트는 단말-노드와 SA를 통해 VOD 서버에 서비스 요청
Step 2: 단말-노드와 SA는 요청된 비디오의 첫 번째 스트림을 포함 이어지는 스트림들이 각각 자신의 버퍼와 다른 인근의 단말-노드에 저장되어 있는가를 조사
Step 3: 이미 저장되어 있을 경우 캐시-히트의 경우 서비스 개시
캐시-미스의 경우 캐시-미스 단계로 이동
Step 4: SA는 VOD 서버로의 연결 해지
Step 5: SA는 비디오 스트림의 순서에 따라 서비스 요청 단말-노드에 스위칭 제어 제공
Step 6: 서비스 요청 단말-노드는 인근 단말-노드에 접속
단말-노드로부터 전송되는 스트림 저장 Step 3으로 이동
Step 7: 단말-노드의 버퍼 부족시 캐시 제거전략 단계로 이동

(그림 5) 캐시-히트 알고리듬

클라이언트들의 서비스 요청에 따른 서비스 개시에 앞서 단말-노드(HEN)와 스위칭-에이젼트가 서버로 동시에 연결되는 이유는 요청된 비디오 스트림들이 분산 단말-노드들에 사전에 저장되어 있지 않았을 경우(캐시-미스)에 단말-노드와 스위칭-에이전트의 제어로 인한 단말간 지연이 추가되는 단점을 해결하기 위함이다.

Step 1: 인근의 단말-노드들로부터 수신한 스트림들을 우선적으로 제거
Step 2: 단말-노드들로부터 수신한 스트림들에 대해 LRU를 적용
Step 3: 단말-노드로부터 전송된 첫 번째 스트림 제거
Step 4: 서버로부터 전송된 스트림들에 대해 LFU 적용
Step 5: 서버로부터 전송된 스트림들에 대해 LRU 적용
Step 6: 서버로부터 전송된 첫 번째 스트림들에 대해 LFU 적용
Step 7: 서버로부터 전송된 첫 번째 스트림들에 대해 LRU 적용

(그림 6) 캐시 제거전략 알고리듬

각 분산 단말-노드들을 통한 클라이언트들의 요청이 빈번할 경우 단말-노드들의 제한된 버퍼로 인하여 이미 저장되어 있는 비디오 스트림들이 버퍼로부터 제거되어야 한다. 따라서 본 논문에서 제안된 웹-캐싱 전략은 서비스가 요청될 경우 일단 비디오의 시작부분인 첫 번째 스트림들은 지연시간을 줄이기 위해 버퍼로부터 마지막으로 제거한다. 또한 다른 단말-노드로부터 수신된 스트림들은 서버로부터 직접 수신한 스트림들에 앞서 제거된다. 이와 함께 LFU(least frequently used)와 LRU(least recently used) 등을 조합하여 적용한다.[9] 이와 같은 캐시 제거 전략은 단말-노드들의 캐시 상태를 스위칭-에이젼트에게 필수적으로 알려야 하며 본 논문에서 사용된 캐시 제거전략을 그림 6에 나타냈다.

4. 분산 Web-caching 성능 분석

4장은 제안된 단말-네트워크(HNET)에서의 분산 웹-캐싱 기술을 사용하여 VOD 서버의 부하 감소는 물론 네트워크 자원의 효율성을 향상시킬 뿐만 아니라 단말간 지연을 최소화하기 위한 제안된 분산 캐싱 기능의 성능분석을 위한 시뮬레이션 결과에 대해 자세히 분석한다. 각 비디오 요

청 패턴은 VOD 서버로부터 제공되어진 비디오의 인기도에 의존한다.

VOD 서버에서 제공되는 N개의 비디오 아이템들의 ID로부터 클라이언트들이 i번째로 인기가 있는 비디오 아이템을 선택하는 확률은 Zipf 분포를 사용한다.[11,12,13] 특정 비디오가 i번째로 선택될 확률 pi는 Z/i이고, Z는 다음으로 나타낼 수 있다.

$$Z = 1 / (1 + 1/2 + 1/3 + \cdots + 1/N)$$

단말-노드들을 통해서 VOD 서버가 서비스를 제공하는 전체 클라이언트들의 서비스 요청 율(request rate)이 λ(분당 서비스 요청 율)일 경우, i번째로 인기가 있는 비디오에 대한 서비스 요청 율은 다음으로 나타낼 수 있다.

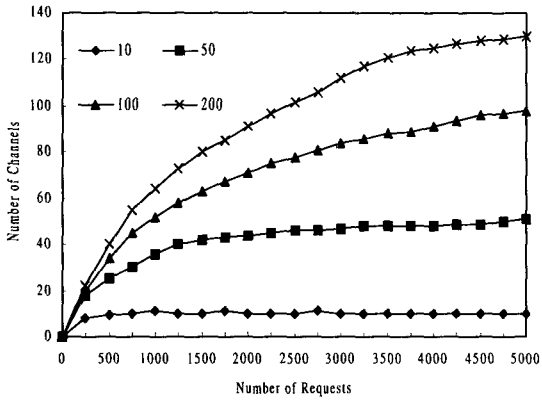
$$\lambda_i = \lambda pi \text{ (여기서 } pi=Z/i)$$

비디오 서버가 제공하는 가장 인기가 있는 비디오(i=1)는 다른 비디오 아이템들에 비해서 보다 더 높은 가중치(weighting)를 두어야 하기 때문에 시뮬레이션에서 각 비디오 선택에 대한 가중치 파라미터로써 pi를 사용한다.[14]

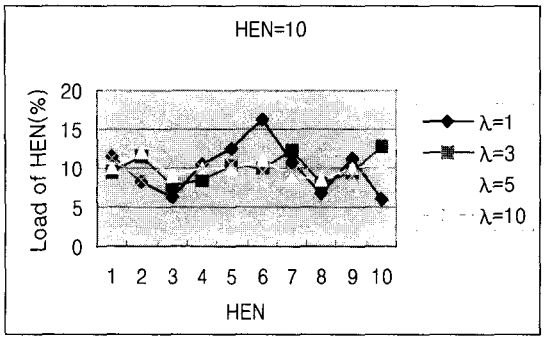
수행된 시뮬레이션 환경하에서 VOD 서버에서 제공된 비디오의 개수(N)는 10개부터 200개까지이고, HNET을 구성하고 있는 각 단말-노드의 서비스 요청 율(λ)은 매분마다 1에서 10까지이다. 예로서 λ=1이란 매분 사용자로부터 서비스 요청이 하나임을 의미하며, 매우 희박한 서비스 요청이 발생되는 경우가 된다. 그러나 비록 서비스 요청율 λ=10이 적은 요청율로 보일지라도, λ=10일 때는 가장 붐비는 경우이다. λ=10에서 100분 동안 전체 요청 율은 1000이 되며, 이 숫자는 각각의 단말-노드마다 전체 가입자의 수를 초과하는 경우가 발생할 수 있다. HNET은 스위칭-에이전트(SA)와 1에서 10개의 단말-노드들로 이루어져 있다. 종단에 위치할 수 있는 각 단말-노드에 장착되는 버퍼의 용량은 다른 단말-노드나 서버로부터 200개 비디오 스트림을 저장한다.

그림 7은 하나의 스위칭-에이전트와 10개의 단말-노드들로 구성된 HNET에서 서버는 10, 50, 100 및 200개의 비디오 컨텐츠를 저장하고 있으며 각 단말-노드들은 200개의 멀티캐스트 비디오 스트림을 저장하는 경우의 필요 전송채널의 수를 보여주고 있다. 사용자들이 요청하는 비디오의 수가 10개부터 200개로 증가하면서 전송에 필요한 채널의 수는 비디오의 증가분보다 적게 증가되는 것을 알 수가 있다. HNET을 구성한 경우 하나의 비디오 컨텐츠는 서버로부터 한번만 전송이 되며 HNET의 각 단말-노드들에 분산 저장된다. 측정된 채널의 수는 서버, 스위칭-에이전트 그리고 각 단말-노드들의 연결 경로 상에서 전송에 사용되는 채널의 평균치를 구한 것이다.

그림 8은 단말-노드에 연결되어 있는 클라이언트들로부터의 서비스 요청 율 λ가 λ=1, 3, 5, 10에 따라서 HNET이 10개의 단말-노드들로 이루어졌을 경우에 대한 각 단말-노드들에 대한 서비스 요청의 부하를 나타내고 있으며, λ=1을 제외하고 단말-노드들의 부하가 균형되어 있음을 나타내고 있다. HNET상의 λ=1에 대해서 균형 있는 부하를 지원하지 않은 이유는 클라이언트들의 서비스 요청이 영화 하나에 대하여 분당 1번 정도의 낮은 요청 빈도수를 갖기 때문이다.



(그림 7) 영화수에 따른 필요 전송채널 수



(그림 8) HNET이 10개의 단말-노드들로 구성된 경우 도착률 변화에 대한 각 단말-노드들의 부하

λ=2 이상의 경우에서 부하가 균형을 이루고 있는 이유는 인기 있는 비디오 서버가 제공하고 있는 인기가 있는 특정 인기 있는 비디오에 대한 각 단말-노드들에 접속되어 있는 클라이언트들의 서비스 요청이 급격하게 증가하기 때문이며, 이로 인해 단말-노드의 캐쉬에는 인기가 있는 비디오가 상대적으로 많이 저장되기 때문이다. 그림 8은 단말-노드의 서비스 요청 율에 대하여 거의 동일한 부하를 갖는 제안된 분산 웹-캐싱 전략의 Equi-loaded 단말-노드를 보여 주고 있다. 또한 이 결과는 HNET을 구성하는 단말-노드들의 수와 별로 관계가 없다는 것을 나타낸다. 특히 λ 가 10에 접근하고 단말-노드의 개수가 10일 때 각 단말-노드의 부하는 거의 10% 정도의 균등한 분포로 발생됨을 확인할 수 있다.

4. 결 론

본 논문에서는 VOD 서버의 전송 채널의 사용을 최소화하기 위한 방법으로서 스위칭-에이전트와 단말-노드들로 구성되는 HNET에서의 웹-캐싱 전략을 제안하였다. 제안된 방식은 VOD 서버로부터의 스트림 전송을 최소화할 수 있으며, 비디오 스트림을 단말-노드들에게 분산 저장하여 스위칭-에이전트의 제어아래 클라이언트들은 단말-노드들로부터 서비스를 제공받기 때문에 서비스 대기시간을 최소화할 수 있다. 또한 특히 인기가 있는 비디오들이 단말-노드들에 균일하게 분산 저장되므로 분산 웹-캐싱에서 큰 문제점으로 대두되는 특정 노드에 대해 부하가 집중되는 단점이 제거 되는 특징을 가질 수 있음을 확인하였으며, 단말-노드의 수가 10개인 경우 각 단말-노드의 부하는 약 10%로 균등하게 분산되고 있음을 확인하였다.

HENT은 스위칭 에이전트의 제어로 동일 비디오에 대해 서비스를 요청하는 클라이언트의 수와는 무관하게 VOD 서버로부터 비디오 당 한 개의 채널만을 사용하여 스트림을 전송 받기 때문에 서버의 입장에서 보면 하나의 HENT은 각 비디오에 대해 하나의 클라이언트로 간주되므로, 동시 사용자의 수는 각 HNET에 구성된 클라이언트들의 합만큼 증대될 수 있다.

참고문헌

[1] T. Little, D. Vnekatesh, "Prospects for Interactive Video-on-Demand," IEEE Multimedia, Fall (1994) 14-23.

[2] R. Rajaie, M. Handley, H. Yu and D. Estrin, "Proxy caching mechanism for multimedia playback streams in Internet," in Fourth International WWW Caching Workshop, Mar. 1999.

[3] C. Shahabi and M.H. Alshayeji, "Super-streaming: a New Object Delivery Paradigm for Continuous Media Servers," Journal of Multimedia Tools and Applications, V11, n1, 275-298, May 2000

[4] L. Fan, P. Cao, J. Almeida and A.Z. Broder, "Summary Cache: A Scalable Wide-Area Web Cache Sharing Protocol," Proceedings of ACM SIGCOMM'98, pp. 254~265. Technical Report 1361, Computer Sciences Department, Univ. of Wisconsin-Madison, Feb. 1998.

[5] R.Rajaie, Haobo Yu, M. Handley and D. Estrin, "Multimedia Proxy Caching Mechanism for Quality Adaptive Streaming Applications in the Internet,"

in Proceedings of IEEE Infocom'2000, Tel-Aviv, Israel, March 2000

[6] S. Sen, J. Rexford and D. Towsley, "Proxy Prefix caching for Multimedia Streams," in Proc. IEEE infocom, 1999

[7] R. Rejaie, M. Handely and D. Estrin, "Architectural Considerations for Playback of Quality Adaptive Video over the Internet," Technical report 98-686, Computer Science Department, USC.

[8] D.S Kim, Y.J.Kim, Iksoo Kim&Yoseop Woo, "VOD Service using a New Web-Caching Technique," 9th IFIP Working Conf. On Performance Modelling and Evaluation of ATM & IP Networks, Budapest, Hungary, pp. 395~401, June, 2001.

[9] H. J. Chen, A. Krishnamruthy, D. Venkatesh, T.D.C. Little, "A Scalable Video-on-Demand Service for the Provision of VCR-like Functions," Proc. 2nd IEEE Int'l. Conf. On Multimedia Computing and System, Washington D.C, May(1995) pp. 65~72.

[10] M. Arlitt and C. Williamson, "Trace-Driven Simulation of Document Caching Strategies for Internet Web Servers," in Simulation Journal, vol. 68, no. 1, pp. 23~33, Jan. 1997.

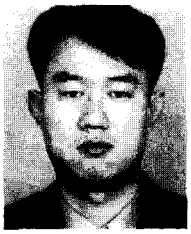
[11] A. Mahanti, C. Williamson, "Web Proxy Workload Characterization," Technical Report Univ. of Saskatchewan, Feb (1999).

[12] B.H.Kim, S.C.Moon, Iksoo Kim&Yoseop Woo, "A Buffering Algorithm for Providing the Truly Interactive Video-on-Demand Services," PDPTA99, June 1999.

[13] P. Cao, L. Fan and G. Philips, "Web Caching and Zipf-like Distributions: Evidence and Implications," IEEE Infocom 1999

[14] Carey Williamson, "On Filter Effects in Web Caching Hierarchies," ACM Transactions on Internet Technology, Vol. 2, No. 1, pp. 47~77, February 2002

◐ 저 자 소 개 ◑



김 백 현
1993년 2월 인천대학교 정보통신공학과 졸업(공학사)
1993년 8월~1997년 12월 삼성전자 전임연구원
2001년 2월 인천대학교 정보통신공학과 대학원 졸업(공학석사)
2000년 12월~2002년 12월 PNP네트워크 선임연구원
2003년 3월~현재 인천대학교 정보통신공학과 대학원 박사과정
관심분야 : Multicast, QoS, 분산처리, 멀티미디어
E-mail : hidesky24@incheon.ac.kr

황 태 준
1997년 2월 인천대학교 전자계산학과 졸업(공학사)
1999년 2월 인천대학교 전자계산학과 대학원 졸업(공학석사)
1999년 3월~2001년 2월 인천대학교 정보통신공학과 대학원 박사과정 수료
관심분야 : Multicast, VOD, Webcaching, 데이터베이스, 멀티미디어
E-mail : tjhwang@incheon.ac.kr

김 익 수
1977년 동국대학교 전자공학과 졸업(공학사)
1981년 동국대학교 전자공학과 대학원 졸업(공학석사)
1985년 동국대학교 전자공학과 대학원 졸업(공학박사)
1988년 3월~현재 : 인천대학교 정보통신공학과 교수
1993년 3월~1994년 2월 North Carolina State Univ. 객원교수
2004년 3월~2005년 2월 California State University Sacramento 객원교수
관심분야 : Multicast, Network, VOD, Webcaching
E-mail : iskim@incheon.ac.kr