특집논문-05-10-3-05

무선 환경에서 QoS 적응적인 스트리밍 프락시 서버를 위한 트래픽 기반 캐싱 알고리즘 및 성능 분석

김 화 성[a)‡], 김 용 술[a)], 홍 정 표[b)]

Traffic-based Caching Algorithm and Performance Evaluation for QoS-adaptive Streaming Proxy Server in Wireless Networks

HwaSung Kim[a)‡], YongSul Kim[a)] and JungPyo Hong[b)]

요 약

멀티미디어 스트리밍 서비스의 증가는 인터넷 컨텐츠의 새로운 국면으로 나타나고 있다. 특히, 무선이동통신망에서 증가하는 멀티미디어 응용에 대한 QoS 제공은 무엇보다 중요하다. 멀티미디어 스트림은 일반적으로 인터넷과 무선망의 유실 특성, 지연, 프로토콜 오버헤드로 인해 높은 초기 지연시간을 가진다. 서비스 제공자는 클라이언트 가까이에 있는 프락시에서 자주 액세스 되어지는 멀티미디어 스트림의 초기 세그먼트를 캐싱함으로써 성능을 향상 시킬 수 있다. 프락시는 서버로부터 스트림의 나머지 부분을 요구함과 동시에 클라이언트에 전송을 시작할 수 있다. 본 논문에서는 캐싱 프락시의 성능을 향상시키기 위해 트래픽 기반 캐싱 기법(TSLRU)을 제안한다. TSLRU는 트래픽을 세 종류로 분류하여 캐싱하며, 교체 대상 결정 시 여러 요소(traffic types, recency, frequency, object size)를 반영함으로 써 캐싱 프락시의 성능을 향상 시킨다. 모의실험에서 캐싱 알고리즘은 byte hit rate과 startup latency에서 높은 성능을 보였다.

Abstract

The increasing popularity of multimedia streaming services introduces new challenges in content distribution. Especially, it is important to provide the QoS guarantees as they are increasingly expected to support the multimedia applications. Multimedia streams typically experience the high start-up delay due to the large protocol overhead, the delay, and the loss properties of the wireless networks. The service providers can improve the performance of multimedia streaming by caching the initial segment (prefix) of the popular streams at proxies near the requesting clients. The proxy can initiate transmission to the client while requesting the remainder of the stream from the server. In this paper, we propose the traffic based caching algorithm (TSLRU) to improve the performance of caching proxy. TSLRU classifies the traffic into three types, and improve the performance of caching proxy by reflecting the several elements such as traffic types, recency, frequency, object size when performing the replacement decision. In simulation, TSLRU performs better than the existing schemes in terms of byte hit rate, hit rate, startup latency, and throughput.

Keywords : Caching, Proxy, Streaming

a) 광운대학교 전자통신공학과
Dept. Electronic and Communications Engineering, KwangWoon Univ.
b) LG 전자 DM 연구소
LG Electronics DM Research Institute

※ 본 연구는 정보통신부 및 정보통신 연구진흥원의 대학 IT연구 센터 육성 지원사업의 연구 결과로 수행되었음

Ⅰ. 서 론

유무선 네트워크 기술의 급속한 발전과 개인용 컴퓨터 장비의 확산으로 사용자는 다양한 형태의 유무선 정보서비스를 접하고 있다. 휴대폰, PDA, 노트북과 같은 개인용 통신 장비는 기존의 텍스트뿐만 아니라 멀티미디

어 데이터를 송수신 할 수 있도록 기능이 향상되고 있고, 이를 바탕으로 다양한 유무선용 콘텐츠가 개발되고 있다. 특히, 동영상을 이용한 스트리밍 서비스가 최근 활성화되고 있다. 이러한 서비스를 가능케 하기 위해서는 동영상 코덱 기술, 멀티미디어 서버 제작 기술, 네트워크 QoS (Quality of Service) 제어 기술 등과 같은 핵심 기술을 필요로 한다. 성공적인 동영상 스트리밍 서비스를 위해서는 종단 간 QoS가 보장되어야 하지만, 사용자의 증가가 기간망의 확장 속도를 추월하고 있으며 결과적으로 트래픽의 급격한 증가로 인한 네트워크 병목 현상이 두드러지고 있다. 이를 해결하기 위해 기간 네트워크의 신속한 확장 및 교체를 고려할 수 있으나, 경제적 비용 부담이 적지 않다. 따라서 현실적인 대안으로 대두되고 있는 방법은 기존 네트워크상에서 작동하는 스트리밍 미디어 캐싱 서비스를 구축하는 것이다[1][2].

스트리밍 미디어 캐싱은 원격 호스트 간에 콘텐츠를 효과적으로 전달할 수 있는 지역 네트워크 중심의 캐싱 서비스 모델이다. 즉, 지역 네트워크상에서 캐시 서버를 전략적으로 배치하고 캐시 적중을 통하여 콘텐츠를 효율적으로 분배하여 사용자 QoS를 향상 시킨다. 이러한 스트리밍 미디어 캐싱 시스템 개발은 기존의 일대일 스트리밍 서비스를 포함한 미디어 캐싱 환경을 지원하기 위해 콘텐츠 캐싱 방식, 캐시의 저장 미디어, 전송 미디어, 전송 프로토콜, 캐시 입출력 시스템과 캐싱 정책을 고려해 개발 되어야 한다.

본 논문에서는 무선 네트워크에서 스트리밍 서비스의 종단 간 QoS 보장을 위한 QoS 적응적인 프락시 서버 아키텍처를 설계하고, 프락시에서 가장 중요한 요소 중 하나인 캐싱 정책을 제안하였다. 캐싱 정책은 콘텐츠 캐싱 및 교체 정책(replacement policy)을 포함하며, 종단 간 QoS 성능 향상을 목적으로 하고 있다. 2장에서는 QoS 적응적인 프락시 서버 아키텍처에 대해 기술하고, 3장에서는 기존의 캐싱 정책과 새롭게 제안하는캐싱 정책에 관해 기술 한다. 4장에서는 본 논문에서 제안한 방식의 실험결과에 대해 설명한다. 마지막으로 5장에서는 결론 및 향후 연구 계획을 설명한다.

Ⅱ. QoS 적응적인 스트리밍 프락시 서버 구조

프락시는 클라이언트 인근에서 인기 있는 콘텐츠를 캐싱함으로써 네트워크 트래픽을 감소시키고 전송 지연을 감소시킬 수 있는 효율적인 기술이다. 현재까지 인터넷에서의 멀티미디어 프락시에 대한 연구가 진행되었으며, 그 대부분은 BHR(byte hit ratio)이나 초기 전송 지연과 같은 확실한 성능 측정 기준의 높고 낮음으로 캐시 교체 정책을 최적화하는 것에 목적을 두고 있었다[3]. 반면 무선 환경에서의 프락시 기법에 대한 연구는 스트리밍 데이터를 중계하기 위한 전송률 제어 모듈이나 포맷 전환을 위한 트랜스코드를 중심으로 이루어 졌다[4][5]. 본 논문에서 제한하는 프락시 역시 인터넷에서의 캐시 교체 정책과 무선망에서의 전송률 제어 모듈을 기반으로 설계되었다. 하지만, 그림 1과 같이 종단간 QoS 성능 향상을 위해 세션 관리, 자원 관리, 이동성 관리, 캐싱 관리 모듈로 세분화 하였다.

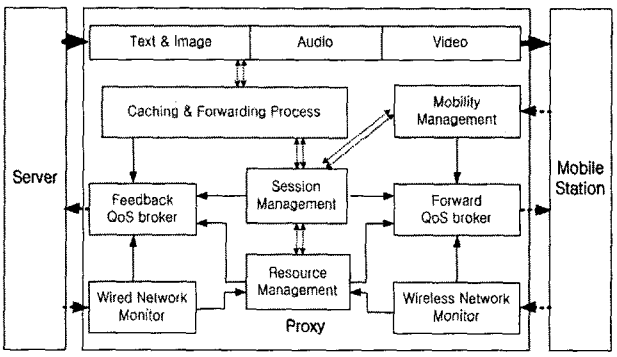

그림 1. QoS 적응적인 스트리밍 프락시 서버 아키텍처
Fig. 1. QoS-Adaptive Streaming Proxy Server Architecture

QoS 적응적인 스트리밍 프락시는 4개의 중요 관리 모듈과 각각 2개의 QoS 중계자, 모니터링 모듈로 구성되어 있다. 모니터링 모듈은 유무선 네트워크에서 발생하는 패킷 유실, 지연에 관한 정보와 무선 네트워크에서 이동 단말의

이동성에 관한 정보를 세션, 자원, 이동성 모듈에 전달하는 역할을 한다. 각 모듈은 전달받은 QoS 파라미터를 기반으로 QoS 정책을 수행한다. 자원 관리 모듈은 유무선 상에서의 적응적인 자원 할당 및 예약을 관장하며, 유선 네트워크에서는 RSVP(resource reservation protocol)을 사용하고, 무선 네트워크에서는 RRNS(resource renegotiation scheme)[6]을 사용한다. 반면 이동성 관리 모듈은 단말의 핸드오프나 이동에 따라 전송률 조절 및 자원 할당을 요청한다.

세션 관리 모듈은 세션 제어, 프로토콜 제어, 표준 멀티미디어 프로토콜로 구성된다. 세션 제어는 상위에 서비스 요구에 대한 서비스 세션을 관리하며 프로토콜 제어는 세션 서비스에 대한 프로토콜 제어와 입출력 요구를 처리한다. 표준 멀티미디어 스트리밍 프로토콜 (RTSP/RTP/SDP)은 스트리밍 미디어 캐싱 서비스에 적용하기 위해 내부적인 기능 확장을 통해서 클라이언트와 서버에 대해서 투명한 전송을 담당한다. 또한 실질적인 QoS 조절 모듈로 전달받은 QoS 파라미터를 이용하여 무선 네트워크로의 전송률을 유동적으로 조절하고 서버와 클라이언트 사이에 생성된 세션을 관리한다.

캐시 관리 모듈은 수용 제어 정책과 캐시 교체 정책으로 구성된다. 수용 제어 정책은 디스크와 네트워크 대역폭의 시스템 자원과 캐시의 저장 공간에 대하여 사용자 제어를 수행한다. 즉 캐시 입출력 시스템의 포화 상태를 방지하고 사용자 QoS를 보장하기 위한 정책이다. 캐시 교체 정책은 저장해야 할 콘텐츠와 제어해야 할 콘텐츠를 선택하는 정책으로 입출력 부하, 사용자 QoS에 대한 캐싱 시스템의 효율성 구조를 논리적으로 제시한다. 캐싱 정책은 본 논문에서 제안하는 트래픽 기반의 스트리밍 캐싱 알고리즘을 사용한다.

Ⅲ. 트래픽 기반의 캐싱 알고리즘

1. 관련연구

캐싱 시스템의 목적은 자주 사용하는 대상 데이터에 대해 사용하는 주체에 근접한 위치에 두어 구성 시스템의 효율성과 사용자 QoS를 향상시키는데 있다. 즉 캐싱 정책은 자주 사용하는 데이터에 대한 플레이스먼트 정책과 사용하지 않는 데이터를 캐쉬에서 제거하는 캐쉬 교체 정책으로 구성되며 그 효율성에 따라 캐싱 시스템의 성능을 결정한다. 일반적으로 캐싱 시스템을 활용하는 분야는 운영체제, 웹이 있으며 스트리밍 미디어에 대한 캐싱은 최근에 연구 활동이 활발해 지고 있으며 다양한 서비스에 대한 사용자 증가로 시제품이 나와 있는 상태이다. 또한 Tertiary 저장 장치에 대한 디스크 캐쉬를 구성하기도 한다. 먼저 운영체제 내에 캐쉬는 버퍼 캐쉬, 페이지 캐쉬 등으로 대변되며 시스템 메모리의 페이지 단위 캐싱을 제공하며 데이터의 지역성(locality)을 높이는 LRU(Least Recently Used), LFU(Least Frequently Used) 정책 등을 사용하고 있다.

웹 캐싱의 캐싱 정책에서는 참조 시간, 참조 빈도, 전송 시간, 오브젝트 크기에 따른 LRU, LFU, SIZE을 기반 정책으로 웹 오브젝트의 특성을 반영한 Hyper-G, Pitkow-Recker, LRU-Threshold, Log(Si-ze)+LRU, Segmented LRU 등이 제시되고 있다[7][8]. Hyper-G 알고리즘은 24시간 동안 참조한 오브젝트를 제외한 나머지 오브젝트에 대해서 LRU을 적용한다. LRU-Threshold는 LRU에 기반 하면서 정해진 크기를 넘어서는 오브젝트에 대해서 캐싱 대상에서 제외한다. Log(Size)+LRU는 Log(Size)값이 가장 크고 최근에 참조되지 않은 오브젝트를 희생 대상을 선택한다. Segmented LRU 역시 LRU를 기반으로 하고 있으며, 그림 2와 같이 프로텍티드 세그먼트(Protected Segment)와 언프

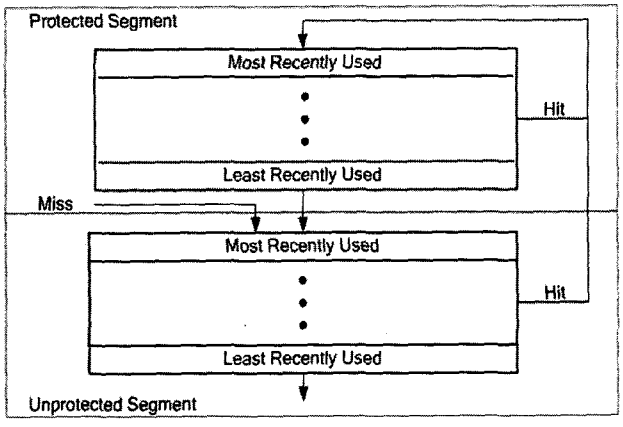

그림 2. 세그먼티드 LRU
Fig. 2. Segmented LRU

로텍티드 세그먼트(Unprotected Segment)의 각각 독립적인 공간에서 LRU를 적용시킴으로써 자주 사용되고 최근에 사용된 오브젝트를 보호하는 방법이다. 그 외에 평균 지연시간을 고려한 Lowest-Latency-First[9], 오브젝트의 인기도와 연관성을 고려한 GreedyDual[10], 그리고 오브젝트의 크기와 마지막 참조 시간을 함수로 표현한 LRV(Lowest Relative Value)[11] 등의 캐싱 정책이 있다.

스트리밍 미디어를 대상으로 하는 캐싱 정책으로는 세그먼트를 이용한 캐싱 정책(SEG)이 있다[12]. 캐싱 된 블록은 세그먼트 단위로 관리하며, 세그먼트 내에 위치하는 블록 수는 i 번째 세그먼트에 2i-1을 할당한다. 이는 미디어 크기가 클수록 대상 미디어에 대한 캐쉬 저장 공간과 교체 정책을 통해 여유 공간을 빠르게 확보하기 위해서이다. 세그먼트 기반의 교체정책은 초기 세그먼트의 개수에 관한 임계값 Kmin 스택과 나머지 세그먼트들에 대한 스택을 통한 이차원 구조를 갖는다. 임계값 Kmin 이하의 세그먼트에 대한 스택은 초기 지연에 대한 사용자 QoS를 보장하기 위한 것이고, 각각의 스택은 LRU 관리를 위해 미디어 ID, 최근에 참조한 시간, 미디어에 대한 마지막 블록 ID로 구성된다. 교체정책은 현재시간(T)과 마지막 참조된 시간(T')의 차이 값(T-T')에 세그먼트의 거리(i)를 곱한 식의 역수(1/((T-T')*i))를 계산하여 작은 값을 갖는 세그먼트를 희생 세그먼트로 선택한다. 즉, 최근에 참조되지 않은 세그먼트 중에 가장 높은 순위의 세그먼트(가장 많은 데이터를 캐싱한 미디어를 선택하여 캐싱 공간을 만든다. 세그먼트에 기반한 정책은 한번의 연산으로 많은 양의 블록을 확보 할 수 있는 반면에 스트리밍 서버로부터 많은 데이터를 받아들여야 하는 점에서 입출력 시스템과 스트리밍 서버에 오버헤드를 갖게 한다. 또한 사용자 QoS를 위한 이차원 LRU 스택은 캐싱되는 미디어 크기가 임계값 Kmin 이하의 세그먼트에 대한 스택에 한정되어 캐싱된 콘텐츠 수의 증가로 적중율이 증가하나 참조량 적중율이 낮은 단점이 있다.

2. 멀티미디어 스트리밍 프락시에 적합한 캐싱 정책

무선 네트워크에서 멀티미디어 데이터(e.g: 스트리밍 서비스, VoIP 등)를 송수신하는 사용자 수의 지수 함수적인 증가와 함께 스트리밍 서버에 가해지는 부하와 인터넷 트래픽도 현저히 증가하여 사용자가 경험하는 접근 지연 시간도 증가하고 있다. 따라서 이들을 효과적으로 줄이는 하나의 방법으로 프락시 캐싱이 널리 사용되어 왔으며, 유한한 크기의 프락시 캐쉬에서 cache hit을 최대화함으로써 사용자의 접근 지연 시간을 최소화하기 위해 효과적인 캐쉬 교체 알고리즘에 관한 많은 연구가 진행되었다. 하지만 기존의 관련 연구들은 알고리즘의 최악의 경우의 성능을 최적화하기 위해 접근요구의 최신성(recency) 또는 동일한 오브젝트에 대한 접근 요구의 발생 빈도(frequency) 중 어느 한쪽만을 고려한 것이 대부분 이였다.

본 논문에서는 무선 네트워크 환경에서 멀티미디어 서비스를 사용자가 좀 더 빠르게 제공받기 위해 그림 3과 같이 캐쉬를 트래픽 별로 분할하여 사용하는 트래픽 기반의 캐싱 알고리즘 (TSLRU : Traffic based Segmented LRU)을 제안하였다. 트래픽 기반의 캐싱 알고리즘은 트래픽을 표 1과 같이 크기 별로 분류하여 각각의 독립적인 공간에 캐싱하며, 각 트래픽 별 캐쉬의 크기는 캐싱 정책의 성능에 큰 영향을 미치기 때문에 주기적으로 각 트래픽의 BHR(Byte Hite Rate)과 HR(Hit rate)를 측정하여 주어진 캐쉬 크기를

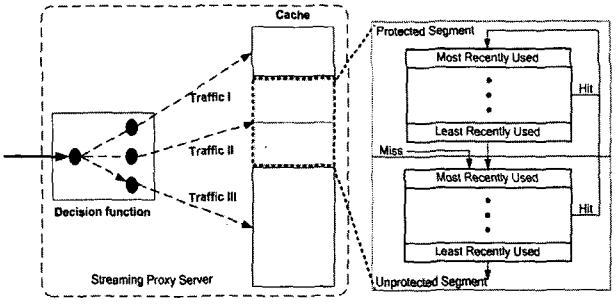

그림 3. 트래픽 기반의 캐싱 알고리즘
Fig. 3. Traffic based Caching Algorithm

표 1. 트래픽 분류
Table 1. Traffic Types

Traffic	Min size	Max size	Example traffic content
I	1 Kbyte	100 Kbyte	Text & Image
II	100 Kbyte	1 Mbyte	Audio streams
III	1 Mbyte	~	Video streams

가변적으로 변경시키는 방법을 사용한다. 예를 들어Traffic 1의 BHR이 20%, Traffic 2의 BHR이 30%, Traffic 3의 BHR이 40%이면 캐쉬를 2:3:4의 비율로 정하게 되며, HR을 사용하여 캐쉬 크기를 조정하는 방법 또한 BHR을 사용한 방법과 동일하다. 또한 나누어진 캐쉬는 독립적인 교체 정책을 사용하게 되며, 교체 대상 결정 시 여러 요소(recency, frequency, size)를 반영할 수 있는 ASLRU (Adaptive Segmented LRU)캐쉬 교체 정책을 사용한다. ASLRU는 SLRU (Segmented LRU)를 기본으로 하고 있지만, SLRU가 고정 크기의 세그먼트를 사용하며 교체대상을 언프로텍티드 세그먼트에서만 결정하는데 반해 ASLRU는 세그먼트들의 크기가 캐쉬 자원 상태에 따라 가변적으로 변하며, 상황에 따라 프로텍티드 세그먼트에서도 교체 대상을 물색한다.

```
US (Unprotected Segment), PS (Protected Segment), c (cache size)

Case I.  request traffic is in US or PS :
        cache hit.
        move request traffic to the top (MRU) of PS.

Case II.  request traffic is neither in US or PS :
        cache miss.
        case (i)  size of US = c / 2 :
                delete the LRU page in US for new traffic.
                insert request traffic the top of US.
        case (ii)  size of US < c / 2 :
                if the cache is full, delete the LRU page in PS for new traffic.
                insert request traffic the top of US.
```

그림 4. 적응적인 세그먼티드 LRU
Fig. 4. Adaptive Segmented LRU

그림 4은 ASLRU 캐쉬 교체 정책을 자세하게 나타낸 것이다. 프락시 캐쉬의 언프로텍티드 세그먼트와 프로텍티드 세그먼트 중 어느 한 곳에서라도 클라이언트가 요구하는 트래픽 오브젝트가 저장되어 있으면 cache hit이 발생하고 그 오브젝트 세그먼트는 프로텍티드 세그먼트의 최상단(MRU)으로 이동된다. 반면, 캐쉬의 세그먼트 두 곳 모두에 트래픽이 저장되어 있지 않으면 cache miss가 발생한다. 이 경우 언프로텍티드 세그먼트의 크기가 전체 캐쉬 크기의 절반과 같으면 새롭게 캐싱 될 오브젝트 세그먼트를 위해 언프로텍티드 세그먼트의 최하단(LRU)에 있는 오브젝트를 지우고 새롭게 캐싱되는 오브젝트 세그먼트를 언프로텍티드 세그먼트의 최상단(MRU)에 추가한다. 만약, 언프로텍티드 세그먼트의 크기가 전체 캐쉬 크기의 절반보다 작고 캐쉬가 가득 차 있는 경우 프로텍티드 세그먼트의 최하단(LRU)의 세그먼트를 지우고 새롭게 캐싱되는 오브젝트를 언프로텍티드 세그먼트에 추가한다. ASLRU는 세그먼트 크기에 따라 적응적으로 교체 대상을 결정하고 가변 크기의 세그먼트를 사용하기 때문에 세그먼트의 크기를 항상 효과적으로 유지하여 SLRU보다 hit rate과 byte hit rate에 있어 좋은 성능을 나타내고, 캐쉬 자원의 활용성을 높이는 장점을 가지고 있다.

Ⅳ. 모의실험 및 성능 분석

캐싱 정책 TSLRU와 ASLRU에 대한 모의실험을 수행하며, hit rate, byte bit rate, startup latency 값으로 성능을 평가한다. Hit rate는 클라이언트의 요청 중 프락시 캐쉬에 존재하는 오브젝트들의 수를 퍼센트로 나타낸 것이고, byte bit rate는 전체 요청 오브젝트의 크기에 대해 요청된 오브젝트 중 캐쉬에 존재하는 오브젝트의 크기를 퍼센트로 나타낸 것이다. Startup latency 는 클라이언트의 초기 요청에서부터 오브젝트의 첫 번째 패킷이 도착할 때까지의 시간을 나타낸다.

1. 캐싱 정책 모의실험 환경

캐싱 정책 모의실험은 그림 5와 같이 하나의 스트리밍 프락시를 가지는 네트워크 모델을 사용하였으며, WebTraff[13] 시뮬레이터에 TSLRU와 ASLRU를 구현하여 실험하였다. 클라이언트로부터 오는 모든 요청은 프락시로 직접 전달되고, 프락시는 요청 파일의 카피본이 존재하는지 확인하여 존재하면 cache hit을 표시하고, 존재하지 않으면 cache miss를 표시하면서 동시에 스트리밍 서버로 파일을 요청한다. 모의실험에 사용된 트래픽은 ProWGen[14]을 사용하여

표 2와 같이 생성하였다. 생성되는 트래픽의 크기는 13 bytes에서부터 53 Mbyte까지 다양하게 하였으며, correlation 값을 zero로 설정하여 트래픽의 크기가 어느 한쪽으로 치우치지 않게 하였다.

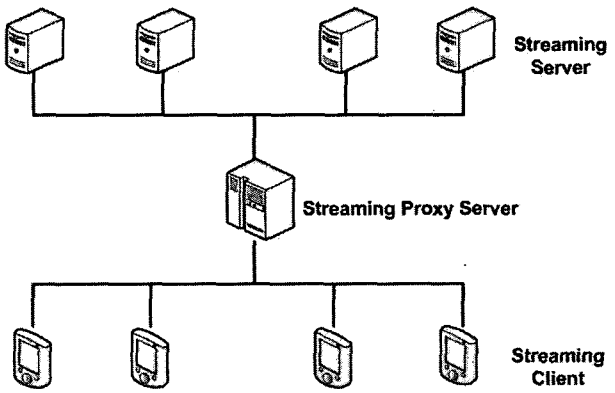

그림 5. 캐싱 정책 모의실험 모델
Fig. 5. Simulation Architecture of Caching Policy

표 2. 트래픽 특성
Table 2. Traffic Characteristics

Item	
Total requests	5,000,000
Unique documents	1,700,000
Unique documents (% of requests)	34%
One-timers	1,224,000
One-timers (% of unique documents)	72%
Total Gbytes of unique documents	19
Smallest file size (bytes)	13
Largest file size (bytes)	53,857,877
Correlation (size and popularity)	zero

2. 캐싱 정책 성능 측정

그림 6는 SLRU, ASLRU, TSLRU-BHR, TSLRU-HR 캐싱 정책의 hit rate를 측정한 그림이다. TSLRU-BHR과 TSLRU-HR는 캐쉬 크기 조정을 BHR이나 HR 을 사용한 것을 나타낸다. 실험 결과, 접근요구의 최신성(recency), 동일한 오브젝트에 대한 접근 요구의 발생 빈도(frequency), 트래픽 크기(size)를 모두 고려한 TSLRU-BHR이 전체적으로는 좋은 성능을 보였지만, 캐쉬 크기가 작을 경우에는 TSLRU-HR이 더욱 좋은 성능을 보였다. ASLRU의 경우 세그먼트의 크기를 캐쉬 크기에 따라 가변적으로 변화시키기 때문에 캐싱 크기가 작을 경우에는 SLRU에 가깝게 나왔으며, 큰 경우에는 TSLRU와 비슷한 성능을 보였다. 또한 캐쉬 크기를 증가하면 증가할수록 hit rate의 성능이 좋아졌지만, hit rate가 65% 정도에 도달한 후에는 일정하게 유지되었다.

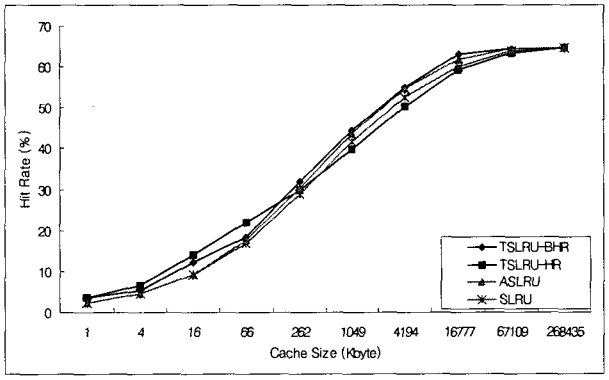

그림 6. 캐쉬 크기에 따른 Hit rate 비교
Fig. 6. Comparison of Hit rate according to Cache Size

그림 7은 캐싱 정책의 byte hit rate를 나타낸 것이다. TSLRU-BHR이 가장 좋은 성능을 보였으며, 그 이유는 각 트래픽이 저장되는 캐쉬의 크기를 각 트래픽의 byte hit rate를 기준으로 비율적으로 정하였기 때문이다. 즉, 캐쉬 크기가 증가하면 할수록 크기가 큰 트래픽이 더 많이 저장

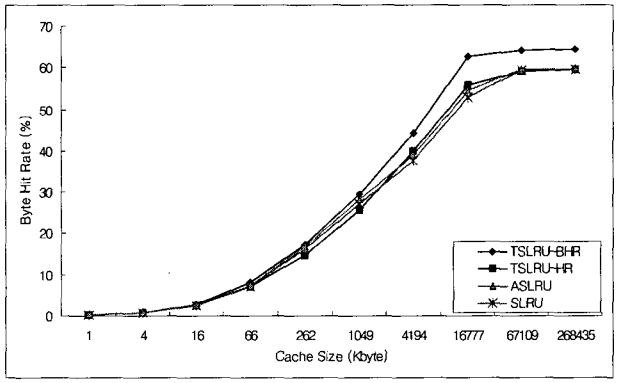

그림 7. 캐쉬 크기에 따른 Byte Hit rate 비교
Fig. 7. Comparison of Byte Hit rate according to Cache Size

되어 그림 12에 나타난 것처럼 비슷한 hit rate인데도 불구하고 byte hit rate 면에서는 높은 성능을 나타내었다. TSLRU-BHR은 65%정도의 byte hit rate 성능을 보였으며, TSLRU-HR, ASLRU, LRU는 60% 정도의 성능을 보임으로써 5% 정도의 성능향상을 나타내었다.

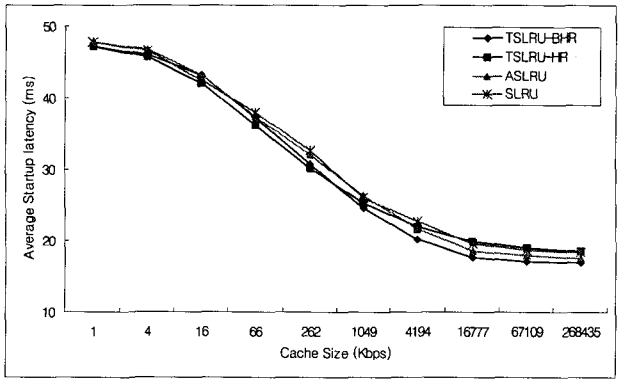

그림 8. 캐쉬 크기에 따른 Startup Latency 비교
Fig. 8. Comparison of Startup Latency according to Cache Size

그림 8은 캐쉬 크기에 따른 캐싱 정책 별 average startup latency를 나타낸 것이다. Startup latency는 hit rate의 영향을 많이 받기 때문에 세가지 캐싱 정책들의 차이가 크게 나타나지 않는다. 하지만, 캐쉬 크기가 작을 경우에는 TSLRU-HR의 startup latency가 낮게 나오고 캐쉬 크기가 커져감에 따라 TSLRU-BHR의 startup latency가 다른 정책들에 비해 낮게 나옴을 알 수 있다. 다시 말해, TSLRU-HR은 멀티미디어 스트리밍 데이터 보다 작은 캐쉬 크기를 유지할 수 있는 텍스트나 이미지 같은 정적인 데이터에서 좋은 성능을 나타내었으며, TSLR-BHR의 경우 캐쉬 크기를 크게 해주면, hit rate 을 다른 캐싱 정책과 비슷하게 유지하면서 데이터 크기가 큰 비디오 스트림에 대한 캐싱 성능을 향상시키며 startup latency 역시 줄일 수 있다.

V. 결론 및 향후 연구과제

본 논문에서는 캐싱 프락시의 성능을 향상시키기 위해 제안한 트래픽 기반 캐싱 기법(TSLRU)은 트래픽을 세 종류로 분류 캐싱하고, 교체 대상 결정 시 여러 요소(recency, frequency, size)를 반영함으로써 캐싱 프락시의 성능을 향상 시켰다.

모의실험에서 캐싱 알고리즘은 hit rate를 기존 캐싱 정책과 비슷하게 유지하면서 byte hit rate와 startup latency에서 높은 성능을 보였다. Byte hit rate의 경우 SLRU보다 5%정도의 성능 향상을 보였으며, startup latency 도 2~3ms 정도 줄어들었다. 하지만 같은 TSLRU방식이라도 BHR을 기반으로 캐쉬 크기를 정한 TSLRU-BHR과 HR을 기반으로 캐쉬 크기를 정한 TSLRU-HR의 성능이 큰 차이를 나타내었다. 즉, 제안한 TSLRU의 경우 성능에 가장 큰 영향을 주는 요소가 각 트래픽들의 가변적인 캐쉬 크기라는 것을 알 수 있었다.

본 논문에서 제안하는 TSLRU는 무선이동통신망에서 멀티미디어 스트리밍 서비스 제공에 있어 기존 방법보다 좋은 QoS 보장 능력을 가지고 있다. 하지만 제안 방안의 성능은 프락시의 전체 캐쉬 크기와 가변적인 트래픽들의 캐쉬 크기에 따라 큰 차이를 보임을 알 수 있었으며, 이를 보완하기 위해서 BHR과 같은 단순한 요소가 아닌 캐쉬와 네트워크 상황에 더욱 적응적인 요소를 통해 각 트래픽의 캐쉬 크기를 관리하고 변화시키는 방법에 대한 연구가 필요할 것이다.

참 고 문 헌

[1] M. Abrams, C.R. Standridge, G. Abdulla, S. Williams, and E.A. Fox, "Caching Proxies: Limitations and Potentials", Proc. of 4th International World Wide Web Conference, pp. 119-133, Boston, Dec 1995.

[2] D.L. Eager, M.C. Ferris, and M.K. Vernon, "Optimized Regional Caching for On-Demand Data Delivery", Proc. 1999 Multimedia Computing and Networking, pp. 301-316, Jan 1999.

[3] Z.L. Zhang, Y. Wang, D. Du, and D. Su, "Video Staging: A Proxy-Server-based Approach to End-to-End Video Delivery over Wide-Area Networks", IEEE/ACM Trans. Networking, Aug 2000.

[4] D.H. Nam, and S. Park, "Adaptive Multimedia Stream Presentation in Mobile Computing Environment", IEEE TENCON 1999, pp. 966-969, 1999.

[5] M. Margaritidis, and G.C. Polyzos, "On the Application of Continuous Media Filters over Wireless Networks", IEEE Intl Conf. on Multimedia and Expo2000, vol. 3, 2000
[6] J.P. Hong, and H.S. Kim, "An Adaptive Resource Allocation Scheme based on Renegotiation Scheme for QoS Provisioning in Wireless Mobile Networks", Proc. of ICOIN2004, vol. 3, pp. 1323-1332, Feb 2004.
[7] S. Williams, M.Abrams, C. R. Standridege, G. Abdulla, and E. A. Fox, "Removal Policies in Network Caches for World-Wide Web Documents," Proceedings of ACM SIGCOMM Conference, August 1996.
[8] R. Karedla, J. Love, and B. Wherry, "Caching Strategies to Improve Disk System Performance," IEEE Computer, Vol. 27, March 1994.
[9] R. Rooster, and M. Abrams, "Proxy Caching that Estimates Page load Delays," Proceedings of the 6th International WWW conference, April 1997.
[10] S. Jin, and A. Bestavros, "GreedyDual Web Caching Algorithm: Expoiting the Two sources of Temporal Locality in Web Request Streams," Proceedings of the 5th international Web Caching and Contents Delivery Workshop, May 2000.
[11] L. Rizzo, and L. Vicisano, "Replacement Policies for a Proxy Cache," IEEE/ACM Transactions on networking, February 1998.
[12] K. Wu, P. s. Yu, and J. L. Wolf, "Segment-based Proxy Caching of Multimedia Streams," Proceedings of the 10th international WWW conference, May 2001.
[13] N. Markatchev, and C. Williamson, "WebTraff: a GUI for Web Proxy Cache Workload Modeling and Analysis," Proceedings of the 10th IEEE International Symposium, October 2002.
[14] M. Busari, and C. Williamson, "ProWGen: a Synthetic Workload Generation Tool for Simulation Evaluation of Web Proxy Caches," ACM Computer Networks, April 2002.
[15] URL:http://www.isi.edu/nsnam/ns/

저 자 소 개

김 화 성

- 1981년 2월 : 고려대학교 전자공학과 졸업
- 1983년 2월 : 고려대학교 전자공학과 석사
- 1996년 : Lehigh Univ 전산학 박사
- 1984년 3월 ~ 2000년 2월 : ETRI 책임 연구원
- 2000년 3월 ~ 현재 : 광운대학교 전자공학부 교수
- 주관심분야 : NGN, 미들웨어 환경, 스트리밍 서비스, 그리딩 컴퓨팅

김 용 술

- 2004년 2월 : 광운대학교 정보통신공학과 졸업
- 2004년 3월 ~ 현재 : 광운대학교 전자통신공학과 석사 과정
- 주관심분야 : QoS-aware 미들웨어 , 스트리밍 서비스

홍 정 표

- 2003년 2월 : 광운대학교 전자공학부 졸업
- 2005년 2월 : 광운대학교 전자통신공학과 석사
- 2005년 2월 : LG전자
- 주관심분야 : Proxy, 무선 스트리밍, 임베디드