모바일 애드 혹 스트리밍 서비스를 위한 프록시 기반 캐싱 최적화

이종득*

Proxy-based Caching Optimization for Mobile Ad Hoc Streaming Services

Chongdeuk Lee*

요 약 본 논문에서는 무선 모바일 애드 혹 네트워크상에서 스트리밍 미디어 서비스 향상을 위한 프록시 기반 캐싱 최적화 기법을 제안한다. 제안된 기법은 WLAN상에서 미디어 서버와 노드들 간의 통신을 위해 프록시를 사용하며, 프록시는 무선 엑세스 포인터 근처에 설치한다. 그리고 캐싱 최적화가 수행되도록 NFCO(non-full cache optimization)과 CFO(cache full optimization)기법을 제안한다. NFCO와 CFO는 프록시에서 스트리밍이 수행될 때 캐시의 성능을 최적화하기 위해 사용된다. 제안된 기법의 성능을 알아보기 위하여 본 논문에서는 제안된 기법과 서버-중심 스트리밍 기법, 그리고 전송율 왜곡 스트리밍 기법과의 성능을 비교 분석하였다. 그 결과 제안된 기법이 서버-중심 스트리밍 기법과 전송율 왜곡 스트리밍 기법에 비해서 최적화 성능이 우수함을 알게 되었다.

주제어 : 무선 모바일, 애드 혹, 캐싱 최적화, WLAN, 프록시

Abstract This paper proposes a proxy-based caching optimization scheme for improving the streaming media services in wireless mobile ad hoc networks. The proposed scheme utilizes the proxy for data packet transmission between media server and nodes in WLANs, and the proxy locates near the wireless access pointer. For caching optimization, this paper proposes NFCO (non-full cache optimization) and CFO (cache full optimization) scheme. When performs the streaming in the proxy, the NFCO and CFO is to optimize the caching performance. This paper compared the performance for optimization between the proposed scheme and the server-based scheme and rate-distortion scheme. Simulation results show that the proposed scheme has better performance than the existing server-only scheme and rate distortion scheme.

Key Words : Wireless Mobile, Ad Hoc, Caching Optimization, WLAN, Proxy

1. 서론

최근에 모바일 애드혹 네트워크상에서 고품질과 고대역폭의 멀티미디어 서비스에 대한 요구가 증가하고 있으나 무선 모바일 네트워크는 유선 네트워크와 달리 많은 제약사항을 가지고 있다. 특히 단말 간 서비스되는 스트리밍 미디어 서비스는 높은 채널 비트 오류율, 채널 페이딩, 그리고 혼잡 (congestion) 등으로 인하여 전송 품질이 저하되고 있다. 이러한 문제를 해결하기 위하여 크로스-레이어 기법, 하이브리드 기법, 그리고 프록시 캐싱 등과 같은 여러 기법들이 제안되고 있다[11][15].

Krishnamachari et al. [4]는 멀티미디어 데이터의 전송 품질을 향상시키기 위하여 적응형 크로스-레이어 보호 기법(adaptive cross-layer protection strategy)을 제안하였다.

그리고 Majumdar et al. [10]은 IEEE 802.11 WLAN상에서 비디오 스트리밍의 확실성을 보장하기 위하여 하이브리드 FEC(Forward Error Correction)/ARQ(Automatic

*전북대학교 전자공학부 교수
논문접수: 2012년 4월 25일, 1차 수정을 거쳐, 심사완료: 2012년 5월 9일

Repeat reQuest) 기법을 제안하였다.

Wang et al. [14]는 ARQ 지연을 줄이기 위하여 프록시 캐싱 최적화 기법을 제안하였으며, [6]과 TranSquid [9]는 캐싱 최적화를 위한 트랜스 코딩 기법을 제안하였다.

프록시 캐싱 최적화 기법은 미디어 전송과 프록시 사이의 트래픽을 줄이고, 클라이언트가 경험한 지연시간을 줄이기 위해 널리 사용되고 있다 [5][8]. 이 기법은 프록시에서 사용빈도가 높거나 관심도가 높은 객체 세그먼트들을 우선적으로 캐싱하여 프록시의 캐싱 오버헤드를 줄인다. 그러나 이 기법은 웹 객체들의 캐싱에는 효율적이지만 미디어 객체들의 경우에는 비효율적인 문제가 발생하고 있다. 이러한 이유는 멀티미디어는 크기 속성으로 인한 캐시의 제약을 받기 때문이다. 또한 무선 네트워크상의 전송율 최적화 관점에서 볼 때 채널 비트 오류율, 채널 페이딩, 그리고 대역폭 제약 등은 스트리밍 전송 품질에 중요한 영향을 끼치고 있다.

따라서 본 논문에서는 모바일 애드혹 네트워크상에서 채널 비트 오류율, 페이딩, 그리고 대역폭 제약 등으로 인한 스트리밍 성능 저하 문제를 개선하고, 스트리밍 미디어 서비스를 최적화하기 위한 프록시 기반 캐싱 기법을 제안한다. 이를 위해서 본 논문에서는 WLAN상에서 클라이언트가 요청한 미디어 스트리밍의 효율성을 위해 프록시와 클라이언트 간의 인코딩율과 디코딩율의 전송율을 분석하여 전송 품질을 최적화한다. 제안된 알고리즘은 모바일 애드 혹 클라이언트의 스트리밍 성능을 최적화하기 위한 것으로서 클라이언트와 프록시 간의 전송율과 원격 미디어 서버와 프록시 간의 전송율을 분석하여 스트리밍의 성능을 측정한다. 일반적으로 클라이언트와 프록시 간의 전송 품질과 원격 미디어 서버와 프록시 간의 전송 품질을 비교해 볼 때 클라이언트와 프록시 간의 전송 품질이 더 우수하다 [6][7]. 이것은 클라이언트가 프록시 근처에서 보다 빠르게 스트리밍을 수행하기 때문이다. 그러나 클라이언트와 프록시 간의 무선 애드 혹 통신은 자원 제약과 물리적 채널 특성 등으로 인하여 전송 품질에 여러 영향을 받게 된다.

이처럼 제안된 메커니즘은 프록시 근처의 모바일 애드 혹 노드들과 스트리밍을 수행하여 최적화를 보장하며, 송신측 버퍼와 수신측 버퍼의 전송율을 조절하여 캐시에서의 혼잡을 최소화한다. 이렇게 함으로써 전송 지연에 따른 패킷 손실을 줄이고, 대역폭 변이(bandwidth variation)로 인한 전송 품질 저하를 최소화한다.

본 논문의 구성은 다음과 같다. 2장에서는 관련 연구에 대해서 살펴보며, 3장에서는 프록시 기반의 스트리밍 최적화 메커니즘에 대해서 살펴본다. 그리고 4장에서는 제안된 메커니즘의 성능에 대해서 살펴보며, 끝으로 결론 및 향후 연구에 대해서 살펴본다.

2. 관련 연구

최근에 모바일 애드 혹 환경에서 스트리밍 미디어 서비스를 위한 여러 캐싱 최적화 기법들이 제안되고 있다. 그러나 무선 애드 혹 환경에서 스트리밍 미디어 서비스 성능을 향상시키기 위한 캐싱 최적화 구조를 설계하는 일은 쉬운 일이 아니다.

이러한 문제를 해결하기 위하여 Nguyen et al. [12]는 각 서버의 전송율을 결정하기 위하여 네트워크 대역폭, 채널 특징, 패킷 손실 등을 고려한 캐싱 최적화 기법을 제안하였다. 이 기법은 패킷들을 분할하여 하나의 서버에 패킷들이 동시에 전송되지 못하도록 제어한 기법이다.

그리고 Hefeeda et al. [2]는 P2P 기반의 미디어 스트리밍을 위해 선택 기법을 이용한 스트리밍 최적화 기법을 제안하였다. 이 기법은 송신측 후보들의 토폴로지와 네트워크 통신 품질을 고려하여 피어들을 선택한 기법으로서 선호도(goodness)에 따라 최상의 송신자가 추론되도록 하였다. 그러나 이들 기법은 WLAN상에서 전송율 및 미디어 객체 크기를 고려하지 않아 전송 효율성 감소와 네트워크 혼잡 문제가 발생하고 있다.

프록시 캐싱 기법은 프록시 캐시에서 미디어 객체들을 스트리밍하기 위한 기법으로서 미디어 객체 세그먼트 전체를 캐싱하는 것이 아니라 세그먼트 일부를 선택해서 캐싱하는 기법이다. 이 중에서 전치캐싱(Prefix Caching) 기법 [15]는 미디어 객체들을 세그먼트하기 위한 초기의 기법으로서 이 기법은 세그먼트를 전치(Prefix)와 후치(suffix)로 구분하여 캐싱을 수행하는 기법이다. Qin et al. [13]은 세그먼트들을 같은 크기로 분할하여 캐싱하는 미디어 객체 스트리밍 최적화 기법을 제안하였으며 이 기법은 클라이언트의 접근 패턴에 따라 세그먼트 길이를 일정한 크기로 분할하여 캐싱을 수행하는 기법이다. 그러나 이 기법은 일정한 크기 분할로 인해 히트율은 증가되나 세그먼트들의 동적 캐싱으로 인하여 오버헤드가 발생하고 있다.

이밖에도 전송율 왜곡 최적화 스트리밍 기법은 미디어 프레임의 스트리밍을 보다 유연하게 서비스해 주는 기법으로서 인코더 버퍼와 디코더 버퍼의 전송율 불일치 문제를 해결하기 위해 사용되고 있다. 이 기법에서 인코더 버퍼와 디코더 버퍼의 전송율 불일치 문제는 스트리밍 최적화에 영향을 미치며, 지터 지연, 네트워크 혼잡 등의 원인을 제공한다.

그리고 Kao et al. [3]은 이득 기반의 트랜스코딩 최적화 기법을 제안하였으며, 이 기법은 단일 경로 상에서 미디어 객체 세그먼트의 이득을 구하여 스트리밍을 최적화한 기법이다. 그러나 이 기법은 프록시 서버와 클라이언트 간의 전체 경로와 이에 따른 미디어 세그먼트의 이득을 미리 알고 있어야 하는 문제점이 발생하고 있다. 따라서 본 논문에서는 이와 같은 문제들을 최소화하고 미디어 세그먼트의 스트리밍 성능을 향상시키기 위해 프록시 캐싱 기반의 스트리밍 최적화 기법을 제안한다.

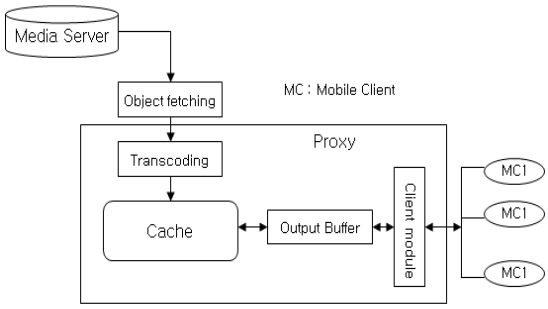
3. 프록시 기반 캐싱 최적화 모델

프록시 기반의 스트리밍 최적화 메커니즘은 네트워크의 고품질을 유지하기 위한 기법으로서 프록시로 하여금 미디어 데이터의 전송 스케줄링을 최적으로 유지하기 위한 것이다.

이 장에서는 캐싱 최적화를 위하여 인코딩-디코딩 불일치 제어 최적화 기법 EDICS (encoding-decoding inconsistency control scheme)을 제안하며, 제안된 기법은 미디어 객체 세그먼트의 인코딩율(encoding rate)과 목적지의 디코딩율(decoding rate)을 모니터링하여 스트리밍을 최적화한다.

3.1 최적화 모델

그림1은 제안된 최적화 모델이며, 미디어 서버, 프록시, 클라이언트, 그리고 스트리밍이 수행되는 세그먼트들의 집합(노드 s, s∈{1, ... ,S(t)}로 구성되어 있다. 여기서 S(t)는 임의의 t시간에 스트리밍이 수행되는 세그먼트이다.



[그림 1] 제안된 스트리밍 최적화 모델

3.1.1 버전 세그먼트

미디어 객체는 한 개 이상의 세그먼트들로 구성되어 있으며, 인코딩을 위해 트랜스코딩을 수행한다. 트랜스코딩이 수행되는 세그먼트 버전들의 비트스트림은 상호 관련성을 유지하며, 본 논문에서는 세그먼트 버전들의 상호 관련성을 모델링하기 위하여 DAG(Directed Acyclic Graph)에 기반을 둔다. DAG에서 각 $version_i$는 세그먼트i를 나타내며, 방향성 에지는 두 객체 세그먼트들 사이의 관련성을 나타낸다. 미디어 서버에서 패치되는 미디어 객체 세그먼트는 스트리밍 최적화를 위한 버전이다. 스트리밍 최적화를 위한 세그먼트 버전 SV_i는 SV_i= {n_i, T_i, T_R}로 구성된다. 여기서 n_i는 스트리밍이 수행되는 미디어 객체 세그먼트의 크기이고, T_i는 미디어 객체 세그먼트를 전송하는 전송시간이다. 그리고 T_R은 소스에서 목적지로 미디어 객체 세그먼트가 전송되는 전송율이다. 따라서 소스에서 목적지로 인코딩율과 디코딩율이 일치성을 이룰 때 미디어 객체 세그먼트는 최적 스트리밍을 수행하게 된다.

3.1.2 전송율 제어

전송율 제어는 혼잡상황과 무선링크 오류로 인한 패킷 손실이 발생할 때 송신율과 전송율을 조절하기 위한 것이다. 본 논문에서는 전송율 제어를 위해 [10]에서 제안된 기법을 이용하였으며 식(1)과 같다.

$$TR_c = \frac{s}{4^* RTT}\left(3+\sqrt{25+24\left(\frac{1-\pi}{\pi-\omega}\right)}\right)\times B \qquad (1)$$

여기서 B는 bytes/s당 전송율, s는 패킷크기, RTT는 round-trip time, ω는 무선링크오류로 인한 패킷 손실율, 그리고 π는 혼잡상황과 무선링크오류로 인한 패킷손실을 포함한 전체 패킷 손실율이다. 이때 송신측은 무선링크 오류로 인해 발생된 불필요한 전송율을 제어하기 위하여 식(1)을 적용한다. 그리고 전체 손실율 π는 수신측에서 측정하며, 무선 손실율 ω는 첫 번째 링크가 무선링크이면 송신측의 MAC계층구조에서 검색된다. 송신측이 모바일 전송이 아닌 경우에는 무선 손실율 ω는 피드백을 통해 송신측으로 이동된다.

3.1.3 최적화 척도

그림1의 스트리밍 최적화 모델에서 노드m과 노드n 사이의 전송채널을 CH_{mn}이라 하자. 여기서 $m,n \in \{0, \dots, S(t), P_c, MC\}$로 정의하며,$m \neq n$, P_c는 프록시 캐시, MC는 모바일 클라이언트이다. 각각의 채널은 한 개의 포워드 경로와 백워드 경로를 가지고 있으며, 이들 경로에서 채널 관리기는 임의의 지연이 발생할 때 지연을 발생시키는 패킷을 관리한다. 전송 채널에서 패킷 손실은 노드m의 인코딩율과 노드n의 디코딩율 사이의 인코딩/디코딩 불일치에 의해 주로 발생된다. 본 논문에서는 CH_{mn}상의 인코딩, 디코딩 불일치로 인해 발생하는 임의의 패킷 손실을 EDI_{loss}(encoding-decoding inconsistency loss)라 하고, 식(2)와 같이 정의한다.

$$EDI_{loss} = 1 - \frac{\omega_i}{\sum_{i=1}^{n} \theta_i} \quad (2)$$

여기서 θ_i는 미디어 객체 세그먼트의 처리율이며, ω_i ($0 \leq \omega \leq 1$)는 m에서 n까지의 포워드 경로에서 응답지연에 따른 손실율이다.

따라서 패킷 손실로 인해 발생된 지연 PL_{delay} (packet loss delay)은 식(3)과 같이 정의된다.

$$PL_{delay} = EDI_{delay-mn} \times \lambda \quad (3)$$

여기서 $EDI_{delay-mn}$은 $EDI_{loss-mn}$에 의한 손실지연계수이며, λ($0<\lambda<1$)는 지연 상수이다.

그러나 소스와 목적지 간의 패킷을 전송하는 과정에 있어서 채널 간섭 등으로 인한 예기치 않는 오류들이 발생할 수 있으며, 이때 오류 최소화는 EM(error minimization)은 식(4)와 같이 정의한다.

$$EM = (1 - EDI_{loss}) \int_0^{T_{begin} - T_{end}} EDI_{delay}(x)\, dx \quad (4)$$

송신측 j와 클라이언트 간의 스트리밍은 송신측 j에서 프록시로 그리고 프록시에서 클라이언트로 구분하여 스트리밍을 수행하게 되며, 각각의 스트리밍을 위해서 손실상수와 지연상수를 적용하여 스트리밍을 최적화한다.

3.2 프록시 최적화

프록시에서 스트리밍 최적화는 미디어 서버와 프록시 간 그리고 프록시와 클라이언트 간에서 수행되는 모든 스트리밍 토큰들을 고려한다. 이 과정에서 프록시 캐시는 지터(jitter)가 발생되지 않도록 해야 하며, PS_i가 프록시 최적화를 위한 스트리밍 토큰일 때 프록시가 요청한 객체 세그먼트 SD_i를 최적화하기 위한 최적화 연산자들은 <표 1>과 같다.

〈표 1〉 프록시 최적화 연산자

연산자	의미
SD_i	세그먼트 데이터
$T_{deadline}^{(k)}$	스트리밍 종료시간
$U_i^{(k)}$	요청자 ID
SL_i	세그먼트 길이
E_r	세그먼트 인코딩율
D_r	세그먼트 디코딩율
B_w	채널 대역폭

따라서 각각의 세그먼트 데이터 SD_i, $i \in \{1, 2, \dots,$

k}에 대하여 스트리밍 최적화 정책π_i은 π_i={$T_{deadline}^{(k)}$, $U_i^{(k)}$, SL_i, E_r, B_w}로 정의한다. 프록시에서 요청자j가 요청한 세그먼트 데이터 SD_i를 스트리밍하기 위한 최적화 정책은 미디어 서버와 프록시간 그리고 프록시와 클라이언트 간으로 구분하여 수행하며, 미디어 서버와 프록시 간의 최적화 전략 OS_{sp}와 프록시와 클라이언트 간의 최적화 전략 OS_{ps}는 식(5), 식(6)과 같이 정의한다.

$$OS_{sp} = \frac{SL_i \times (D_r - E_r)\, T_{deadline}^{(k)}}{B_w} \quad (5)$$

$$OS_{ps} = (n+1)SL_i \times T_{deadline}^{(k)} - \frac{SL_i \times (D_r - E_r)}{B_w} \quad (6)$$

이처럼 스트리밍 최적화는 프록시 지터를 최소화하여 전송 품질을 향상시키기 위한 것이다.

3.3 캐싱 최적화

프록시에서 캐시의 크기가 유한적이고, 캐시가 풀(full)일 때 SD_i 패킷을 전송하면 지연이 발생되거나 충돌로 인한 혼잡이 발생한다. 이러한 문제를 피하기 위해서는 프록시 캐시의 상태를 모니터링 하여 패킷을 조절해야 하며, 본 논문에서는 캐시가 풀이 아닐 때의 캐시 최적화 $NFCO$(non-full cache optimization)와 캐시가 풀일 때의 캐시 최적화 CFO(cache full optimization)로 구분하여 캐싱 최적화를 수행한다.

3.3.1 $NFCO$에 의한 최적화

$NFCO$에 의한 최적화는 캐시가 풀이 아닐 때 캐시의 히트율을 최적화하기 위한 것이다. 본 논문에서는 캐시 히트율 최적화를 위해 요청자가 요청한 SD_i 패킷을 포워딩하기 전에 클라이언트 캐시에 SD_i 패킷 사본을 미리 저장하여 포워딩을 수행한다. 이것은 포워딩 과정에서 발생하는 패킷 손실 최소화, 캐시에서의 수신 오류 최소화, 그리고 혼잡으로 인한 데이터 플로우를 최소화하기 위한 과정이다. 따라서 저장된 패킷 사본은 클라이언트에서 SD_i 패킷을 직접 포워드하기 위해 사용된다. 그리고 캐시가 풀이 아니면 프록시는 캐시로부터 직접 SD_i 패킷을 패치하여 스트리밍을 수행한다.

프록시에서 캐시 크기는 제한적이기 때문에 일부 SD_i 패킷은 캐시되고, 그리고 다른 일부는 클라이언트 캐시에서 대기상태로 스케줄링 된다. 그러나 만일 재전송이 수행되는 SD_i 패킷이 클라이언트 캐시에서 대기중일 때는 프록시는 클라이언트에게 SD_i 패킷을 직접 포워딩하도록 요청한다. 이제 캐시가 풀이 아닐 때 SD_i 패킷은 i번째의 시도를 통해서 프록시 캐시에 성공적으로 포워드 된다고 가정하자. SD_i 패킷이 성공적으로 캐시에 포워드 되면, 다음 SD_i 패킷 포워딩을 위해 이전 SD_i 패킷 전송 이력과 사본 정보는 무시되며, 이전 패킷 정보는 프록시에 저장되게 된다. 따라서 캐시가 풀이 아닐 때 deadline 시간 T_i이전에 요청된 SD_i 패킷이 캐시에 히트될 히트율 δ는 식(7)과 같이 정의된다.

$$\delta = 1 - \left(\frac{EDI_{loss} \times PL_{delay} \times EMO}{\sum_{i=1}^{n} SL_i} \right) \times \alpha \quad (7)$$

여기서 α는 캐시 이용률을 의미한다.

3.3.2 CFO에 의한 최적화

CFO에 의한 최적화는 SD_i 패킷의 중요도에 따라 최적화를 수행하는 기법이다. 즉 캐시가 풀일 때 프록시 캐시를 파악하여 중요도가 낮은 순으로 SD_i 패킷을 제거하기 위한 것이다. 그리고 난 후 CFO는 스트리밍을 요청한 SD_i 패킷의 순위를 다시 결정하게 되며, 순위가 높은 순으로 스트리밍 큐를 재구성하게 된다.

CFO에서 SD_i 패킷 실행은 실행을 준비 중인 패킷이 캐시가 풀 상태 인지를 먼저 파악한 후 풀 상태이면 프록시는 요청자에게 SD_i 패킷을 대기하도록 하고, 이 경우 자신의 클라이언트 캐시에서 스트리밍 패킷을 직접 스케줄링 한다. 그러나 스트리밍을 수행할 캐시가 풀 상태가 아니면 스트리밍은 클라이언트에서 프록시로 스케

줄링 된다. 따라서 클라이언트에서 요청한 SD_i 패킷이 캐시 풀 상태이면, deadline 시간 T_i시간 후 최적화를 수행하게 되고, T_i시간 후 요청된 SD_i 패킷이 캐시에 히트 될 히트율 δ는 식(9)와 같이 정의한다.

$$\delta=1-(T_{end}-T_{begin})\times \left(\frac{EDI_{loss}\times PL_{delay}\times EMO}{\sum_{i=1}^{n} SL_i}\right)\times\alpha\times\beta \qquad (8)$$

여기서 β는 SD_i 패킷의 중요도 계수이며, $0\le\beta\le1$이다.

이와 같이 캐싱 최적화는 캐시 풀 상태에 좌우되며, 캐시 풀을 효율적으로 제어하고 스케줄링할 때 스트리밍 성능은 향상되게 된다.

4. 시뮬레이션 분석

본 논문에서는 이벤트-지향 시뮬레이션을 통하여 성능을 분석하였다. 성능 분석을 위해 4.1절에서는 시뮬레이션 환경에 대해서 살펴보며, 그리고 4.2절에서는 시뮬레이션 결과들에 대해서 살펴본다.

4.1 시뮬레이션 환경

본 논문에서는 시뮬레이션을 위해 무선LAN과 유선LAN이 결합된 유/무선 네트워크 환경을 고려하였다. 네트워크 토폴로지는 하나의 미디어 서버, 하나의 AP, 그리고 10 개의 클라이언트 노드로 구성하였다. 프록시 노드는 AP와 상호 작용하여 위치를 공유하도록 하였으며, IEEE 802.11 상에서 2.4GHz로 모바일 노드들과 통신을 수행하도록 하였다. 그리고 서버는 유선통신상에서 프록시/AP와 통신을 수행하도록 하였으며, 유선통신의 경우 포워드와 백워드의 PLR(Pack Loss Rate)은 모두 3% 이내로 하였다. 서버와 프록시 노드 간의 링크 대역폭은 10/100Mbps로 설정하였으며, 무선 링크의 BER(Bit Error Rate)는 1.2×10^{-5}로 설정하였다. 그리고 링크 레이어에서의 전송 지연은 20μs 이내로 설정하였으며 링크 레이어에서의 최대 재전송 패킷 수는 3개 이내로 제한하였다. 무선 채널의 데이터 전송율은 10Mbps로 제한하였으며, 프록시와 클라이언트 간의 채널 대역폭은 1.5Mbps로 제한하였다. 시뮬레이션 데이터로는 뉴스 비디오에서 캡처한 미디어 객체 세그먼트를 소스로 이용하였다. 전체 시뮬레이션 시간은 560s 동안 수행하였으며, 스트림의 t_s는 [1, 20s], μ는 $\mu\ge0.7$, α는 $0<\alpha<1$로 설정하였다. 연속적인 스트리밍 패킷 데이터는 10개로 제한하였으며, 최대 PSNR(Peak Signal to Noise Ratio)은 30dB이내로 제한하였다. 따라서 시뮬레이션을 위해 사용된 성능 파라미터는 <표 2>와 같으며, 서버로부터 미디어 객체들을 패치하는 데 걸리는 지연은 고려하지 않았다.

〈표 2〉 성능 파라미터

Parameters	value
세그먼트 데이터 수	6,000
세그먼트 길이	512K
패킷 손실율	3% 이내
링크 대역폭	10/100Mbps
최대 비트 오류율	1.2×10^{-5}
링크 레이어에서의 전송지연	20μs
최대 재전송 패킷 수	3
채널 대역폭	1.5Mbps
μ	$\mu\ge0.7$
α	0<0<1
최대 PSNR	30dB 이내
연속적인 스트리밍 패킷 데이터	10개 이내

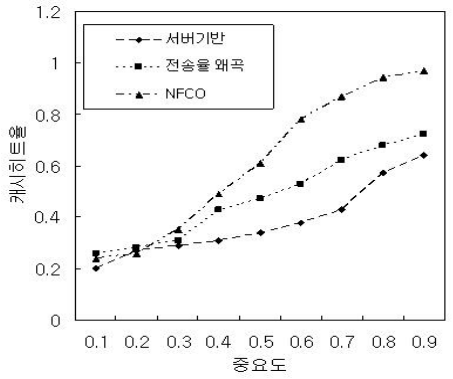
그리고 캐싱 최적화의 성능 일관성을 위해 원본 서버로부터 미디어 객체를 패치하는 데 걸리는 지연은 무시하였다.

4.2 시뮬레이션 결과

성능 평가를 위해서 본 논문에서는 표2의 성능 파라미터를 반영하여 시뮬레이션을 수행하였다. 시뮬레이션에서 사용된 주요 성능 척도는 NFCO에 따른 캐시히트율, CFO에 따른 캐시 히트율, 캐시 이용율에 따른 패킷 손실율이다. 따라서 이들 성능 척도에 기반을 두고서 시뮬레이션을 수행하였으며, 성능 비교는 서버 기반 기법, 전송율 왜곡 기법, 그리고 제안된 기법으로 구분하여 수행하였다.

4.2.1 NFCO에 의한 캐시 히트율

첫 번째 성능평가는 NFCO에 따른 캐시 히트율이다. 아래 그림에서처럼 중요도가 0.5 이하일 때는 버퍼 캐시 히트율에 영향을 미치게 됨을 알 수 있다. 이에 대한 시뮬레이션 결과는 [그림 2]와 같다.

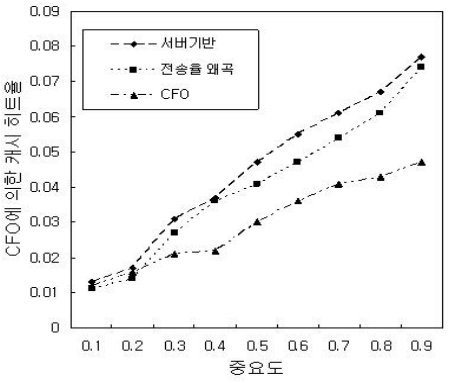


[그림 2] 중요도에 따른 NFCO 캐시 히트율

[그림 2]에서 보듯이 제안된 NFCO 캐시히트율은 비교적 우수한 전송율 왜곡 기법에 비해서 중요도가 0.7이상일 때 버퍼 캐시 히트율이 약 25% 정도 향상되었음을 알 수 있었다. 이것은 제안된 기법에서 스트리밍 미디어 객체들에 대해서 캐싱 최적화가 적용되었기 때문이다. 따라서 스트리밍 미디어 객체 크기를 버퍼 캐시에 최적화 할 때 버퍼 캐시가 효율적으로 제어됨을 알 수 있다. 그러나 적합성 척도 관점에서 볼 때 중요도는 버퍼 캐시에 영향을 미치게 된다. 만일 중요도가 0.6 이하일 때는 미디어 서버로부터의 패치 지연이 크게 발생하기 때문에 버퍼캐시의 성능이 떨어지게 된다. 본 논문에서는 이와 같은 현상을 예방하기 위하여 스트림 데이터가 너무 작거나 캐시 용량을 초과하는 미디어 스트림 데이터들에 대해서는 최적화율을 적용하여 캐시 히트율이 향상되도록 하였다.

4.2.2 CFO에 의한 캐시 히트율

두 번째 성능평가는 CFO에 따른 캐시 히트율이다. [그림 2]처럼 [그림 3] 또한 중요도가 0.5 이하일 때는 버퍼 캐시 히트율에 영향을 미치게 됨을 알 수 있다. 이에 대한 시뮬레이션 결과는 [그림 3]과 같다.

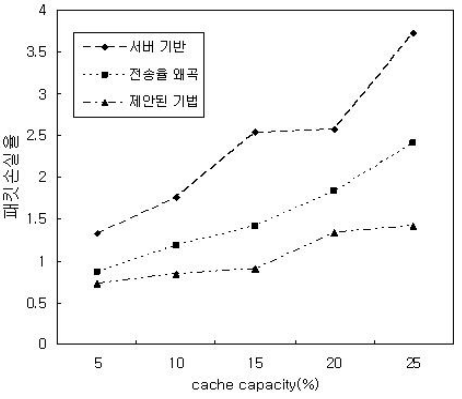


[그림 3] 중요도에 따른 CFO 캐시히트율

[그림 3]에서 보듯이 중요도가 증가할 때 제안된 기법의 캐시 히트율이 향상됨을 알 수 있다. 이것은 중요도가 낮은 순으로 스트리밍 데이터 패킷을 제거하기 때문이다. 결과적으로 스트링 데이터 패킷은 CFO 최적화에 영향을 받는 다는 것을 알 수 있다. 이처럼 중요도가 낮으면 스트림 패킷 데이터들에 대한 참조가 분산되기 때문에 캐시 히트율 또한 낮아지게 된다. 따라서 제안된 기법은 각 스트림 데이터 패킷에 대한 중요도가 적용되었기 때문에 스트림 데이터 패킷 수가 증가하거나 크기가 증가할 때도 다른 기법들에 비해서 캐시 히트율 성능이 향상되게 된다.

4.2.3 캐시 이용율에 따른 패킷 손실율

세 번째 성능평가는 캐시 용량에 따른 캐시 이용율에 따른 패킷 손실율이다. 패킷 손실율 평가를 위해 캐시 이용율을 5%에서 25%까지 변화시켜가면서 성능을 분석하였다. 이에 대한 결과는 [그림 4]와 같다.



[그림 4] 캐시 이용율에 따른 패킷 손실율

[그림 4]에서 보듯이 제안된 기법은 패킷 손실율 관점에서 볼 때 다른 기법들에 비해서 성능이 향상됨을 알 수 있다. 특히 비교적 성능이 우수한 전송율 왜곡 기법에 비해서 캐시 이용율이 20% 정도일 때 약 5%의 성능이 향상되었음을 알 수 있다. 그러나 서버기반 기법의 성능이 상대적으로 낮은 이유는 스트림 데이터 패킷의 크기 및 캐시 용량에 중요도와 최적화 상수를 적용하지 않았기 때문이다. 특히 멀티미디어 객체는 웹 객체와는 달리 프록시 서버에서 지연으로 인한 패킷손실이 더 크게 발생된다. 이와 달리 제안된 기법은 스트리밍 미디어 객체에 캐싱 최적화가 적용하였기 때문에 캐시에서 크기가 큰 스트림 데이터 패킷이 새로 입력되거나 버퍼 캐싱이 수행될 때 패킷 손실을 효과적으로 줄일 수 있는 장점을 제공하게 된다.

5. 결론

인코더 버퍼와 디코더 버퍼의 전송율 불일치 문제는 캐싱 최적화에 중요한 영향을 미친다. 본 논문에서는 스트리밍 서비스를 최적화하기 위한 캐싱 최적화 기법을 제안하였다. 제안된 기법은 최적화 모델을 구성하여 캐싱 최적화가 수행되도록 하였으며, 인코딩과 디코딩 불일치로 발생하는 패킷 손실을 최소화하기 위하여 EDI_{loss} 함수를 제안하였다. EDI_{loss}함수는 패킷 손실로 발생하는 지연을 최소화하며, 간섭, 혼잡으로 인한 오류를 최소화한다. 그리고 캐시 풀 상태에 따라 최적화가 적응성 있게 수행되도록 NFCO 캐싱 최적화와 CFO 캐싱 최적화 기법을 제안하였다. 이 기법은 포워딩 과정에서 발생하는 패킷 손실, 지연 오류, 그리고 혼잡을 최소화한다. 따라서 제안된 캐싱 최적화 기법은 캐시 구조를 안정적으로 제어하고 스트리밍 스케줄링을 효율적으로 수행하게 된다. 시뮬레이션 결과 중요도가 0.9일 때 캐시 히트율의 경우 제안된 기법은 기존의 서버 기법, 전송율 왜곡 기법에 비해서 각각 30%, 25%의 성능 향상을 보였으며, 패킷 손실율은 서버 기법, 전송율 왜곡 기법에 비해서 약 30%의 성능 향상을 보였다. 향후 연구로는 이동성을 고려한 캐싱 최적화 기법에 대한 연구를 고려중이다.

참 고 문 헌

[1] Acharya, S., Korth, H., and Poosala. V.(1999). Systematic Multiresolution and its application to the World Wide Web. in Proc. IEEE ICDE'99, Sydney, Australia, 40-49.

[2] Hefeeda, M., Habib, A., Botev, B., Xu, D., and Bhargava., B. (2003). Promise: Peer-to-Peer Media Streaming using Collectcast. in ACM International Conference on Multimedia, Berkeley, California, USA, 1-10.

[3] Kao, C. F., and Lee, C. N. (2007). Aggregate Profit-Based Caching Replacement Algorithms for Streaming Media Transcoding Proxy Systems. IEEE Transactions on multimedia, 9(2), 221-230.

[4] Krishnamachari, Van der Schaar, Choi, S. M. S,. and Xu. X.(2003). Video Streaming over Wireless LANs: A Cross-Layer Approach. in Proceedings of Packet Video Workshop, Nantes, France, 1-9.

[5] Lee, C. D., Jeong, T. W. (2010), Fuzzy Filtering Based Segment Grouping for User-Centered Multimedia Streaming Service in P2P Distribution Mobile Networks. Journal of Internet Technology, 11(5), 651-658.

[6] Lee, C. D. (2011). Object version Transcoding for streaming media service in Wireless Mobile Networks. KONI, 15(3), 355-363.

[7] Lee, C. D. (2011). Multi-level streaming using Fuzzy Similarity in P2P Distribution Mobile Networks. KONI, 15(3), 364-371.

[8] Li, D., Chuah, C. N., Cheung, G., and Ben Yoo,, S. J. (2005). MUVIS: Multi-source Video Streaming Service over WLANs. KICS, 7(2), 144-156.

[9] Maheshwari, A., Sharma, A., Ramamrithan, K., and Sheonoy. P. (2002). Transquid: Transcoding and Caching Proxy for Heterogeneous e-commerce Environments. in Proc. IEEE INFOCOM 2002, San Jose, CA, 50-59.

[10] Majumdar, A. D. G., Sachs, I. V., Kozintsev, K., and Yeung. M. M. (2002). Multicast and Unicasy Real-time Video Streaming over Wireless LANs. in IEEE Transactions on Circuits and Systems for

Video Technology, 12(6), 524-534.
[11] Miao, Z., Ortega, A. (1999). Proxy caching for Efficient Video Services over the Internet. In Packet Video Workshop, 1-21.
[12] Nguyen, T., and Zakhor, A.(2002). Protocols for Distributed Video Streaming. in IEEE ICIP, Rochester, NY, USA, 1-4.
[13] Qin, M., and Zimmermann, R. (2010). An Adaptive Strategy for Mobile Ad Hoc Media Streaming. IEEE Transactions on Multimedia, 12(4), 317-329.
[14] Wang, T., Fang, H., and Chen. L.(2002). Low-delay and Error-robust Wireless Video Transmission for Video Communications. in IEEE Transactions on Circuits and Systems for Video Technology, 12(12), 1049-1058.
[15] Wu, K. L. Yu, S. and Wolf, J. L.(2004). Segmentation of Multimedia Streams for Proxy Caching. IEEE Transactions on Multimedia, 6(5), 770-780.

이 종 득

・1983년 2월 : 전북대학교 컴퓨터과학과(이학사)
・1989년 2월 : 전북대학교 컴퓨터과학과(이학석사)
・1998년 2월 : 전북대학교 컴퓨터과학과(이학박사)
・1992년 3월~2002년 2월 : 서남대학교 컴퓨터통신학과 교수
・2002년 2월~2012년 5월 현재 : 전북대학교 전자공학부 교수
・관심분야 : 무선 모바일 네트워크, 무선센서 네트워크, MIMO, 유비쿼터스 통신 등