분산 트랜스코딩 환경에서 부하 균형을 위한 트랜스코딩 부하 예측 기법

(Transcoding Load Estimation Method for Load Balance on Distributed Transcoding Environments)

서 동 만 † 허 난 숙 † 김 종 우 † 정 인 범 ††

(Dongmahn Seo) (Nansook Heo) (Jongwoo Kim) (Inbum Jung)

요 약 최근 무선통신 기술의 발전으로 PC뿐만 아니라 PDA, 휴대폰 등 다양한 이동 단말 장치를 통하여 멀티미디어 서비스를 제공받을 수 있게 되었다. 이동 단말 장치는 하드웨어의 성능 제약이 있으며, 낮은 네트워크 대역폭을 가지는 무선망에서 동작한다. 이러한 이동 단말 장치의 특성을 고려한 스트리밍 미디어 서비스를 받기위해서는 동작 환경에 적합하게 미디어를 트랜스코딩 기술이 필요하다. 미디어에 대한 트랜스코딩은 트랜스코딩 서버들에서 이동 단말기 등급별로 수행되어 스트리밍 미디어의 실시간 전송 요구사항에 맞추어 사용자에게 보내져야한다. 대규모의 이동 단말 사용자들 각각에 맞는 QoS의 트랜스코딩 스트리밍 미디어를 제공하기위해서는 트랜스코딩 서버들의 부하분배 정책에 서버에서의 트랜스코딩 부하를 반영하는 것이 필요하다. 본 논문에서는 분산 트랜스코딩 환경에서의 부하 균형을 위한 트랜스코딩 서버에서의 트랜스코딩 부하를 예측 기법을 제안한다. 제안된 기법은 트랜스코딩 서버 정보와 영화 정보, 목적 트랜스코딩 비트율을 이용하여 예상 트랜스코딩 시간을 예측한다. 예측된 시간은 실험을 통하여 실제 트랜스코딩 시간과 유사함을 확인한다.

키워드 : 트랜스코딩, 부하 균형, MPEG, 시간 예측

Abstract Owing to the improved wireless communication technologies, it is possible to provide streaming service of multimedia with PDAs and mobile phones in addition to desktop PCs. Since mobile client devices have low computing power and low network bandwidth due to wireless network, the transcoding technology to adapt media for mobile client devices considering their characteristics is necessary. Transcoding servers transcode the source media to the target media within corresponding grades and provide QoS in real-time. In particular, an effective load balancing policy for transcoding servers is inevitable to support QoS for large scale mobile users. In this paper, the transcoding load estimation algorithm is proposed for load balance on the distributed transcoding environments. The proposed algorithm estimates transcoding time from transcoding server information, movie information and target transcoding bit-rate. The estimated transcoding time is proved based on experiments.

Key words : transcoding, load balance, MPEG, time estimation

· 본 연구는 지식경제부와 한국산업기술재단의 지역혁신인력양성사업으로 수행된 연구결과임

† 학생회원 : 강원대학교 컴퓨터정보통신공학과
dmseo@snslab.kangwon.ac.kr
nsheo@snslab.kangwon.ac.kr
jw_kim@snslab.kangwon.ac.kr
†† 종신회원 : 강원대학교 컴퓨터정보통신공학과 교수
ibjung@kangwon.ac.kr
(Corresponding author임)

논문접수 : 2008년 3월 31일
심사완료 : 2008년 7월 15일

정보과학회논문지: 시스템 및 이론 제35권 제10호(2008.10)

1. 서 론

최근 멀티미디어와 정보통신망의 발전에 따라 영상 정보 서비스에 대한 요구가 날로 다양해지고 있다. 멀티미디어 서비스의 급속한 발전으로 사용자는 유선망 이외에 무선망을 통하여 무선 이동 단말로 스트리밍 미디어를 전송하고 재생하는 서비스를 받을 수 있게 되었다. 그러나 스트리밍 미디어 서비스를 위해서는 영상 정보의 양이 텍스트 기반의 데이타 정보량에 비하여 매우 크기 때문에 광대역 네트워크 대역폭 및 고성능의 컴퓨터를 필요로 하고 있다[1-5].

무선망에서는 네트워크 대역폭이 유선망보다 상대적으로 열악한 환경을 가지고 있으며, 이동 단말의 낮은 컴퓨팅 파워와 시스템 자원은 서버로부터 전송되는 높은 품질의 스트리밍 미디어를 실시간으로 처리하기 어렵다. 이러한 문제점을 해결하기 위하여 최근에 스트리밍 미디어를 이동 단말에 적합한 품질로 바꾸는 트랜스코딩 기술이 연구되고 있다[4,6,7]. 트랜스코딩은 멀티미디어 컨텐츠를 최초 인코딩한 형태에서 목적하는 단말에 적합하게 변환하는 기술이다. 변환 종류로는 단말 사양에 맞게 스트리밍 미디어의 프레임율, 해상도, 디스플레이 크기, 비트율의 조절을 포함하여 MPEG 1, 2 미디어를 MPEG-4 미디어로 변환하는 요구도 포함한다[4,5].

트랜스코딩 시스템은 인코딩한 영상 데이타를 가지고 있는 멀티미디어 서버와 영상 데이타를 단말 환경에 맞게 변환하는 작업을 수행하는 트랜스코딩 서버로 구성된다. 이동 단말에서 스트리밍 미디어 서비스 요청을 하게 되면 멀티미디어 서버에 저장중인 영상 미디어를 사용자 단말 환경에 적합한 형태로 변경하기 위하여 트랜스코팅 서버로 전송한다. 트랜스코딩 서버는 전송한 영상 미디어를 이동 단말 환경에 적합한 형태의 스트리밍 미디어로 바꾸어 스트리밍 서비스를 한다. 트랜스코딩은 많은 CPU 자원을 요구하는 작업으로, 일반적인 트랜스코딩 시스템에서 트랜스코딩 서버가 단일로 구성되는 경우 동시 접속 사용자 수가 제한된다. 또한 짧은 시간에 대규모 사용자로부터 트랜스코딩 작업이 요청되면 사용자의 QoS가 보장되는 기간 안에 작업을 처리하기 어렵다. 대규모 트랜스코딩 서비스에서는 단일 서버로 부하가 집중되는 문제를 해결하기 위해 여러 대의 트랜스코딩 서버로 시스템을 구성하여 트랜스코딩 작업 요청을 분배 할 수 있다[4-7].

대규모 분산 트랜스코딩 시스템에서는 단일 서버로 부하가 집중되는 문제를 해결하기 위하여 서버 간에 트랜스코딩 작업을 균형적으로 분배하는 것이 중요하다. 미디어 데이타의 경우 각 데이타의 크기가 일정하지 않으며 비트율, 프레임율 변경 등 트랜스코딩 하는 방식에 따라 작업 시간의 차이가 발생하기 때문에 미디어 데이타의 정보와 각 트랜스코딩 서버의 자원 정보를 이용하여 트랜스코딩 작업 부하를 정확하게 예측하고, 예측한 정보를 통하여 트랜스코딩 서버를 동적으로 선택한다면 특정 트랜스코딩 서버로 작업 부하가 집중되는 현상을 방지 할 수 있다.

본 논문에서는 분산 트랜스코딩 환경에서의 부하 균형을 위해 트랜스코딩 작업 시간 예측 기법을 제안한다. 제안하는 기법은 미디어 데이타의 원본 비트율과 트랜스코딩 목적 비트율, 서버의 자원 정보를 이용하여 트랜스코딩 작업 시간을 예측한다. 예측한 트랜스코딩 작업 시간은 실험을 통하여 실제 트랜스코딩 작업 시간과 유사한 특성을 가짐을 보인다.

본 논문의 구성은 다음과 같다. 2장에서는 본 논문의 관련 연구에 대하여 설명한다. 3장에서는 MPEG-2 미디어 데이타의 트랜스코딩 과정을 분석하여 트랜스코딩 작업 시간 예측 알고리즘을 제안한다. 4장에서는 제안된 알고리즘을 통해 예측한 트랜스코딩 작업 시간과 실험을 통해 측정한 트랜스코딩 작업시간을 비교 분석한다. 마지막으로 6장에서는 본 논문의 결론을 맺고 향후 연구 계획을 설명한다.

2. 관련 연구

2.1 MPEG 압축 표준

MPEG(Moving Picture Experts Group)은 대표적인 음성과 영상 압축 표준 중 하나로 압축 알고리즘의 차이에 따라 MPEG-1, 2, 4 등으로 분류된다. 그 중 MPEG-2는 MPEG-1의 기능을 개선 및 확장한 압축 표준이다. MPEG-2는 5~10Mbps 이상의 비트율로 MPEG-1에서 보다 높은 화질을 실현하기 위해 개발되었으며 방송 분야에서의 응용을 주목적으로 한다. 방송용 영상을 위해 새로운 목표 비트율을 설정하고, 비트율을 높여 비월주사 방식에도 대응할 수 있도록 하거나, 움직임이 빠른 화면에서 고화질을 얻는 동시에 정지 영상에서도 고해상도를 얻을 수 있도록 필드/프레임 적응 부호화 방식을 채택하여, 영상의 성질에 따라 VLC 테이블을 교체하는 연구를 진행하여 개발되었다.

MPEG-1, 2의 PS(Program Stream)는 비디오 시퀀스(Video Sequence) 계층과 GOP 계층, 그리고 픽처(Picture) 계층으로 구성되어 있다. 비디오 시퀀스 계층은 일련의 같은 속성을 갖는 화면 그룹으로서 화면 크기, 화면 비율 등의 기본적인 정보를 가지고 있다. GOP 계층은 랜덤 액세스의 단위가 되는 화면 그룹의 최소 단위로 영화를 상영하는 기본 단위이다. 픽처 계층은 화면 한 장의 공통된 속성으로 하나의 프레임과 동일하다.

픽처 계층에서 픽처는 기능적으로 서로 다른 I, P, B,

D 네 종류의 타입을 가진다. I 픽처는 자신의 화면 정보만으로 부호화되는 화면으로, 다른 프레임과는 상관없이 독립적으로 화면에 나타날 수 있기 때문에 랜덤 액세스를 위해 하나의 GOP 내에 최소 1개 이상의 I 픽처가 필요하며 I 픽처에서부터 영화가 상영된다. P 픽처는 I 픽처 또는 다른 P 픽처로부터의 예측을 수행함에 따라 생기는 화면이다. 일반적으로 P 픽처에는 순방향 픽처 간 예측 정보를 담고 있다. B 픽처는 MPEG의 특징인 쌍방향 예측에 의해 생기는 화면으로, 일반적으로 과거 영상으로부터 예측하는 순방향 프레임 간 예측 정보와 미래로부터 예측하는 역방향 프레임 간 예측 정보 그리고 전후 양방향으로부터의 예측에 의한 정보를 담고 있다. D 픽처는 VCR 기능을 지원하기 위한 픽처이나 실제 거의 사용되지 않는다[8-10].

2.2 MPEG 압축 기술

DCT(Discrete Cosine Transformation)는 공간적인 화상을 저주파 성분과 고주파 성분의 주파수별로 분해하는 것으로 영상 압축을 위한 준비 작업이다. 저주파는 이웃하는 픽셀들과 색차나 밝기의 차이가 거의 없거나 적고 고주파는 차이가 많은 것을 의미한다. 보통의 영상은 통계적으로 고주파보다 저주파가 많기 때문에 DCT를 수행할 경우 대부분의 값들이 저주파에 집중된다. 상대적으로 고주파에는 값이 거의 존재하지 않기 때문에 고주파 부분의 값을 무시해도 영상에 큰 영향을 미치지 않는다. DCT는 8×8 크기의 블록을 기반으로 이루어지며 DCT 과정을 통해 얻어진 결과 값을 DCT 계수라고 한다. 일반적으로 DCT 계수는 실수 값으로 그대로 저장할 경우 많은 저장 공간을 필요로 하기 때문에 높은 압축 효율을 얻을 수 없다[8,9].

양자화(Quantization)는 DCT 계수 값에 대하여 실제적인 데이타 압축을 수행한다. 지그재그 스캔(Zigzag Scan), 대체 스캔(Alternative Scan)과 같이 영상의 화질에 영향을 미치는 정도를 고려하여 일정하게 정의된 순서에 의해 DCT 계수를 읽어 들이면 비슷한 특성을 가지는 값들이 연속적으로 나타나게 된다. 이 값들을 미리 정해지거나 사용자에 의해 정의된 양자화 계수로 나눔으로써 일정한 구간으로 재편성하고 정수로 변환한다. 이 때 양자화 계수가 클수록 압축률이 높아지고 반대로 양자화 계수가 작을수록 압축률이 낮아진다. 따라서 양자화 계수에 따라서 해당 미디어 데이타의 비트율이 결정되며, 압축률이 높을수록 낮은 비트율의 미디어 데이타가 가변길이부호화 단계에서 생성된다[8,9].

가변길이부호화(Variable Length Coding)는 높은 확률로 발생하는 값을 짧은 코드로, 낮은 확률로 발생하는 값은 보다 긴 코드로 지정함으로써 데이타 흐름의 전체 비트 수를 감소시키는 부호화 방법이다. 가변길이부호화(VLC) 과정에서는 가상 버퍼를 만들고 양자화 결과 값을 읽어 들여 가변길이부호화를 수행하는데 이 때 가상 버퍼에 발생하는 비트의 양에 따라 비트율을 제어한다[8,9].

2.3 트랜스코딩 시스템

일반적인 트랜스코딩 시스템의 구조는 그림 1과 같다. 사용자는 트랜스코딩에 필요한 정보를 트랜스코딩 서버에 전송 한다. 트랜스코딩 서버에서는 요구한 스트리밍 미디어의 원본을 미디어 서버에서 읽어 사용자가 요구한 해상도, 비트율, 프레임율에 따라서 트랜스코딩한 후 사용자에게 전송한다. 예를 들면 트랜스코딩 서버로부터 CIF(352×288)등급의 25프레임/초, 비트율 100Kbps의 비디오 스트리밍을 QCIF(176×144)등급의 15프레임/초, 비트율 50Kbps의 스트리밍으로 사용자에게 전송할 수가 있다. 트랜스코딩 서버에서는 무선망의 특성에 적합하도록 비트율, 해상도, 프레임율을 변경 할 수가 있다.

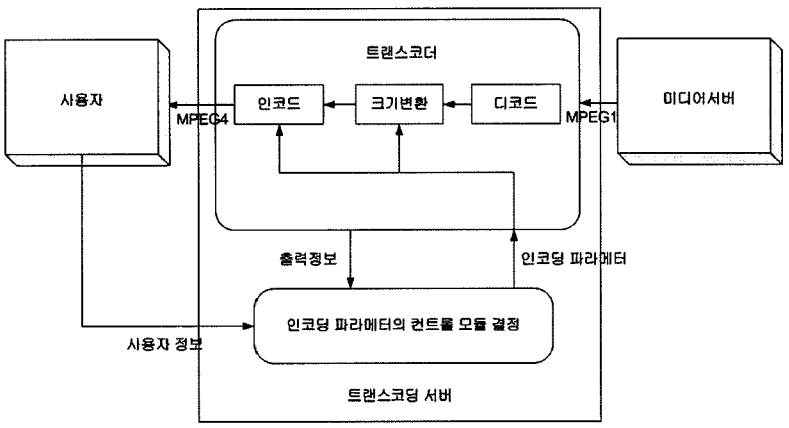

그림 1 트랜스코딩 시스템 구조

기존의 트랜스코딩 시스템을 구축하는 방법들로 소스기반 정적 인코딩 시스템과 트랜스코딩 서버 시스템이 있다[4,11]. 그림 2는 소스기반 정적 인코딩 시스템 방식을 보여준다. 이 방식은 사용자의 요구에 대하여 미디어 파일들을 미리 사용자 등급별로 트랜스코딩한 후 서버에 저장하여 사용하는 방식이다. 이러한 시스템은 스트리밍 미디어를 실시간으로 트랜스코딩하여 전송하는 방식보다 서버의 부하가 심하지 않다는 장점이 있으나 무선 랜 환경에서 사용자의 이동에 의한 네트워크 변화에 적응할 수 없다는 단점이 있다. 또한 모든 미디어 파일들을 각 등급별로 트랜스코딩하여 각각의 다른 미디어 파일들로 저장해야 한다는 부담이 있다.

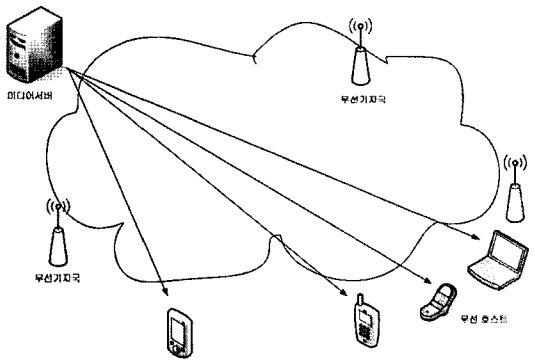

그림 2 소스기반의 정적 인코딩 시스템

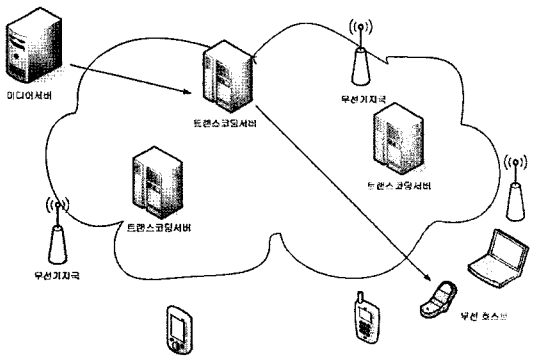

그림 3 정적 트랜스코딩 서버 시스템

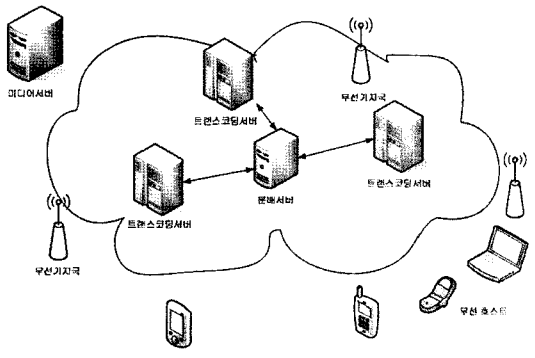

그림 4 부하분배 트랜스코딩 시스템

그림 3은 정적 트랜스코딩 서버 시스템을 보여주고 있다. 이 방식은 이동 단말을 사용하는 사용자와 연결된 무선기지국에서 가장 가까운 트랜스코딩 서버를 선택하고 스트리밍 서비스를 받는다. 그러나 트랜스코딩 서버와 이동 단말 사이의 네트워크 상태 변화에 따라 적절하게 서비스 하지 못하는 단점이 있다.

정적 트랜스코딩 서버 방식의 문제점인 특정 서버에 부하가 집중되는 부하 불균형을 피하기 위하여 분배 서버를 두고 트랜스코딩 서버들의 부하 상태를 파악하여 트랜스코딩 요구들을 처리하는 부하 분산 트랜스코딩 시스템 방식이 있다. 그림 4에 나타나듯이 이 방식에서는 분배서버는 트랜스코딩 서버들과 연동하여 정해진 부하분배 정책에 따라서 트랜스코딩 서버를 선택하게 된다. 전통적으로 클러스터 시스템에서는 부하분배 정책으로 RR, WRR, DWRR, WLC, RWLD 알고리즘 등이 사용되고 있다[4,6,7].

2.4 부하분배 정책

라운드로빈 방식은 사용자 요청이 들어오면 순서대로 서버에 할당하는 방식으로, 부하 간 우선순위의 개념이 없고 서버의 연결 개수나 응답 시간 등은 고려하지 않는다. 스케줄링의 기초는 사용자 기반이며 실제 서버들의 상태들을 고려하지 않기 때문에 효과적인 부하 분배를 기대하기 힘들고 서버 사이에 부하 불균형이 심각해지는 문제가 발생할 수 있다[4,6,7].

최소 접속 방식은 가장 접속이 적은 서버로 요청을 연결하는 방식이다. 각 서버에서 동적으로 실제 접속한 숫자를 세어야 하므로 동적인 스케줄링 알고리즘 중 하나이다. 비슷한 성능의 서버로 구성된 시스템에서는 부하가 아주 큰 요청이 한 서버로만 집중되지 않기 때문에, 데이타의 크기가 작고 사용자 접속 부하가 매우 큰 경우에 효과적으로 부하 분배할 수 있다. 그러나 미디어 스트리밍과 같이 긴 접속을 요구하는 사용자 요청과 비교적 짧은 접속을 요구하는 사용자 요청이 섞여있는 동작 환경에서는 서버들 간의 부하 불균형을 초래할 수 있다. 특히 성능이 서로 다른 이종 노드들로 구성된 클러스터 서버 환경에서는 LC같이 사용자 접속 숫자로 부하 배분을 수행하는 것은 서버들의 비효율적인 사용 결과가 발생하기 쉽다[4,6,7].

가중치기반 라운드로빈은 실제 서버에 서로 다른 가중치를 설정하여 부하 분배에 사용하는 방식이다. 각 서버에 가중치를 임의로 부여할 수 있으며, 여기서 지정한 가중치 값을 통해 처리 순서를 정한다. 예를 들어 기본

가중치가 1이고 서버 A, B, C가 각각 4, 3, 2의 가중치를 갖는 경우 스케줄링 순서는 ABCABCABA가 된다. 가중치기반 라운드로빈을 사용할 경우 서버의 연결 개수를 셀 필요가 없고 동적 스케줄링 알고리즘보다 스케줄링 부담이 적다. 그러나 사용자 요청이 폭주할 경우 라운드로빈과 마찬가지로 서버 사이에 심각한 부하 불균형이 발생할 수 있다[4,6,7].

가중치기반 최소 접속 방식은 최소 접속 방식의 한 부분으로서 서버 간의 성능 가중치를 부여하여, 가중치가 높은 서버에서 더 많은 사용자 요청을 받아들일 수 있도록 한다. 가중치기반 최소 접속 방식은 최소 접속 방식에 비해 가중치에 따라 부하를 배분하는데 부가적인 비용이 발생한다[4,6,7].

자원 가중치 기반 부하분배 정책은 트랜스코딩 서버의 실제 자원 소모량을 측정하고, 이를 기준으로 등급별 자원소모량 가중치 및 각각의 트랜스코딩 서버별로 처리할 수 있는 등급별 총 자원가중치 값을 바탕으로 트랜스코딩 서버들의 부하분배를 수행한다[4].

2.4 트랜스코딩 작업 시간 예측 기법

트랜스코딩 작업 시간 예측 기법은 원본 미디어 데이타의 정보와 목적 트랜스코딩 비트율을 이용하여 작업시간을 예측한다. 이 기법에서는 원본 미디어 데이타의 재생시간과 비트율, 프레임율, 목적 트랜스코딩 비트율을 기반으로 하여 트랜스코딩 작업 시간을 예측한다[12]. 본 기법은 트랜스코딩 작업 시간을 예측하여 데이타를 프리패칭하기 위하여 연구되었으며, 단일 서버에서 트랜스코딩 작업 시간을 측정하여 이를 기반으로 앞서 언급한 데이타를 대입하여 공식을 유도하였다. 따라서 이 기법을 다양한 환경의 트랜스코딩 서버를 가지는 분산 트랜스코딩 환경에 적용하기 어렵다. 본 논문에서는 트랜스코딩 서버의 프로세서의 클럭과 원본 미디어 데이타의 정보, 목적 트랜스코딩 비트율등을 이용하여 분산 트랜스코딩 환경에서의 부하분배 정책에 적합한 트랜스코딩 작업 시간 예측 기법을 제안한다.

3. 트랜스코딩 작업 시간 예측

3.1 MPEG-2 미디어의 비트율 변경 트랜스코딩 과정

트랜스코딩은 원본 미디어 데이타를 입력받아 목적하는 형태로 변환하는 작업을 수행하여 출력한다. 가장 단순한 트랜스코딩 방법은 원본 미디어 데이타를 픽셀 단위까지 완전하게 디코딩 한 후 사용될 목적에 적합하도록 비트율, 프레임율, 해상도 등을 변경한 후 다시 인코딩한다. 이러한 트랜스코딩 방법을 픽셀 도메인(Pixel-domain) 트랜스코딩이라 한다. 픽셀 도메인 트랜스코딩은 미디어 데이타를 픽셀 단위까지 완전하게 디코딩한 후 재인코딩 함으로써 높은 CPU 처리량을 요구하며, 처리해야 하는 데이타의 양이 크기 때문에 많은 메모리 용량을 필요로 한다. CPU와 메모리 등의 시스템 자원의 활용 면에서 보다 효율적인 트랜스코딩 방법으로 미디어 데이타를 움직임 벡터(MV: Motion Vector)와 DCT 계수, 양자화 된 DCT 계수 등의 변환하고자 하는 작업에 직접적으로 영향을 미치는 압축 단계까지만 디코딩하여 비트율과 프레임율, 해상도 등을 변경하고 인코딩하는 압축 도메인(Compressed-domain) 트랜스코딩이 있다. 본 절에서는 픽셀 도메인과 압축 도메인 방식을 이용하여 MPEG-2 미디어 데이타의 비트율을 변경하는 트랜스코딩의 과정에 대하여 알아본다.

그림 5는 픽셀 도메인 트랜스코딩 과정을 나타낸다. 입력된 MPEG-2 미디어 데이타는 첫 번째로 가변길이 복호화(VLD : Variable Length Decoding) 과정을 거쳐 양자화 된 DCT 계수와 MV 같은 매크로블록 단위의 데이타 등 인코딩 과정에서 가변길이부호화(VLC)되기 전의 데이타 형태로 디코딩된다. 다음으로 역양자화(IQ : Inverse Quantization) 과정을 거쳐 양자화 되기 전의 DCT 계수 형태로, 역 DCT(IDCT : Inverse DCT) 과정을 통해 비트스트림 단위의 데이타로 디코딩된다. 픽셀 도메인 단계까지 완전히 디코딩 된 원본 미디어 데이타는 매크로블록마다 목적 비트율에 따라 MV와 인코딩 모드 결정(Mode Decision)을 새롭게 계산하고 다

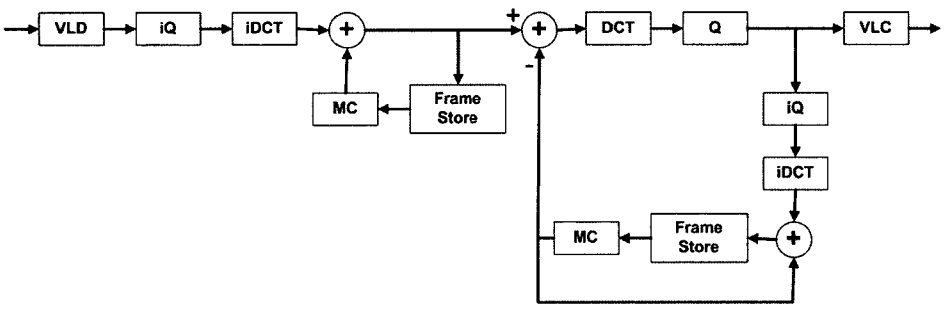

그림 5 픽셀 도메인 트랜스코딩 과정[13]

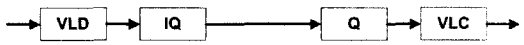

그림 6 Open-loop 방식의 압축 트랜스코딩 과정[13]

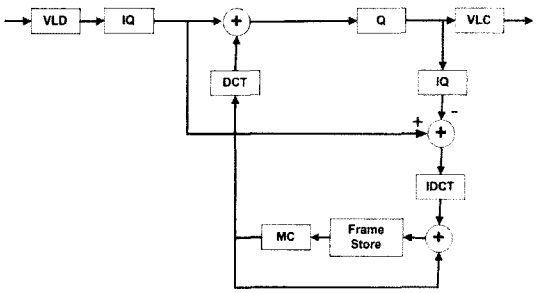

그림 7 Closed-loop 방식의 압축 트랜스코딩 과정[13]

시 DCT와 양자화, 가변길이부호화 과정을 통해 재인코딩을 수행한다[13].

압축 도메인 트랜스코딩은 양자화 된 DCT 계수, DCT 계수 등 인코딩 중간 과정에서 발생하는 압축 데이타 단위로 원본 미디어 데이타를 디코딩 한 후 비트율을 조절하여 다시 인코딩한다. 이때 원본 미디어 데이타를 디코딩하는 수준에 따라 Open-loop와 Closed-loop 방식으로 나눌 수 있다. 그림 6과 그림 7은 각각 Open-loop와 Closed-loop 방식의 트랜스코딩 과정을 보여준다. Open-loop 방식은 가변길이복호화와 역양자화 과정을 통해 원본 미디어 데이타를 양자화 된 DCT 계수 단위로 디코딩하며 Closed-loop 방식은 그보다 한 단계 더 들어가 역DCT까지 수행하여 양자화 되지 않은 DCT 계수로 디코딩 된 원본 미디어 데이타는 목적 비트율에 따라 양자화 계수를 재설정하여 양자화를 수행함으로써 비트율을 변경할 수 있다. 다음으로 DCT와 가변길이부호화 과정을 통해 목적 비트율로 변환된 MPEG-2 미디어 데이타로 압축한 결과를 볼 수 있다[13].

3.2 비트율 변경에 따른 트랜스코딩 작업 시간 예측

본 절에서는 앞에서 분석한 비트율 변경 트랜스코딩 과정을 바탕으로 비트율 변경 트랜스코딩에 소요되는 작업 시간을 예측할 수 있는 알고리즘을 제안한다. 트랜스코딩 작업 시간은 트랜스코딩 각 과정의 수행 시간을 합한 것으로, 표 1의 기호를 이용하여 수식으로 정리하면 식 (1)과 같이 표현할 수 있다. 가변길이복호화(VLD)와 역양자화(iQ), 역DCT(iDCT) 과정을 거쳐 미디어 데이타를 매크로블록 단위까지 완전하게 디코딩한 후 ME/MC(Motion Estimation / Motion Compensation)와 샘플링(Sampling) 등으로 MV와 매크로블록 정보를 재조정하고 DCT와 양자화, 가변길이부호화(VLC) 과정을 통해 다시 인코딩하는 일반적인 트랜스코딩 과정의 작업시간이다.

$$Tr_T = VLD + iQ + iDCT + MC + Sampling + ME/MC + DCT + Q + VLC \quad (1)$$

그러나 본 논문에서는 비트율을 변경하는 트랜스코딩의 작업 시간만을 고려한다. 프레임율과 해상도 등의 변경이 없기 때문에 움직임 차이값 등의 매크로블록의 정보를 재조정하는 과정이 생략되므로 ME/MC와 샘플링 등의 과정에 소요되는 시간은 0으로 처리할 수 있다. 또한 DCT와 양자화는 8×8 크기의 블록 단위로 수행되는 과정으로 비트율의 높고 낮음에 관계없이 동일한 양의 연산을 수행하므로 비트율 변경에 따른 트랜스코딩 시

표 1 논문에서 사용된 기호 및 의미

기 호	의 미	단 위
TrT	트랜스코딩 작업 수행에 소요되는 시간	sec
TrR	변환 비트율에 따라 상대적으로 소요되는 트랜스코딩 작업 수행 시간	sec
SourceBitrate	미디어 데이타의 원본 비트율	bps
TargetBitrate	트랜스코딩 작업을 통해 변환할 비트율	bps
DiffBitrate	원본 비트율과 트랜스코딩 목적 비트율의 차이값	bps
VLC, VLD	가변길이부호화, 가변길이복호화에 소요되는 시간	sec
Q, iQ	양자화와 역양자화에 소요되는 시간	sec
DCT, iDCT	DCT와 역DCT에 소요되는 시간	sec
MC	Motion Compensation에 소요되는 시간	sec
ME	Motion Estimation에 소요되는 시간	sec
Sampling	Sampling에 소요되는 시간	sec
WriteTime	1Byte의 데이타를 메모리에 쓰는데 소요되는 시간	sec
CPUClock	트랜스코딩 서버의 CPU Clock	Hz
playtime	미디어 데이타의 재생시간	sec
inst	putAC 함수를 호출할 경우 실행되는 명령어의 개수	개
UnitOfWrite	putAC 함수에서 데이타를 파일에 쓰는 최소 단위	bit
STT	원본 비트율과 동일하게 트랜스코딩 수행할 때 소요되는 시간	sec

간에 영향을 주지 않는다. 따라서 비트율만 변경하는 트랜스코딩의 작업 시간은 비트율 변경에 직접적으로 영향을 주는 가변길이부호화(VLC)와 그 외 과정의 작업 시간의 합으로 식 (2)와 같이 표현할 수 있다.

$$Tr_T = VLC + \alpha \quad (2)$$
$$(\because \alpha = VLD + iQ + iDCT + DCT + Q)$$
$$(\because MC + Sampling + MB/MC = 0)$$

가변길이부호화(VLC)는 양자화 결과 값을 읽어 들여 엔트로피 부호화함으로써 압축을 수행하는 과정으로 MPEG-2 인코더에서는 런 랭스 코딩(Run-Length Coding)과 허프만 코딩을 이용한다. 양자화 된 결과 값을 지그재그 또는 대체 스캔 방법을 통해 읽어 들이면 비슷한 수준의 양자화 결과 값들이 서로 모이게 되는데 이때 비트율이 낮을수록 0인 양자화 결과 값이 연속적으로 나타나는 빈도가 높아진다. 런 랭스 코딩을 이용하여 0이 아닌 양자화 결과 값이 나타났을 때 이전까지 읽어 들인 양자화 결과 값을 파일에 쓰기 위하여 허프만 코딩을 이용한다. 표 2는 MPEG-2 인코더의 가변길이부호화 과정 중 런 랭스 코딩에 대한 유사코드이다[9, 10]. 8×8 크기의 블록 단위로 양자화가 수행되기 때문에 한 양자화 결과 값은 블록 당 저주파에 해당하는 1개의 DC 계수와 고주파에 해당하는 63개의 AC 계수로 발생한다. 이때 DC 계수는 이전 블록의 DC 계수와 차분부호화를 통해 별도로 처리되고 나머지 63개의 AC 계수가 런 랭스 코딩을 통해 처리된다. run 변수를 이용하여 연속적으로 0값을 가지는 AC 계수가 나타나면 putAC 함수를 호출하여 이전까지의 데이타를 부호화한다. 변환할 비트율이 높아질수록 연속적으로 0값을 가지는 AC 계수의 출현 빈도가 낮아지고, putAC 함수를 호출하는 횟수가 많아지기 때문에 변환할 비트율이 낮을 때 연속적으로 0 값을 가지는 AC 계수의 출현 빈도만 계산하는 것보다 상대적으로 더 많은 작업 시간을 소모하게 된다.

표 2 MPEG-2 인코더의 가변길이 부호화 알고리즘

```
Algorithm VLC {

    /* DC 계수 */
    putDClum(dc_value);

    /* AC 계수 */
    run = 0;
    for(n=1; n<64; n++) {
        if(ac_value != 0) {
            putAC(run, dc_value)
            run = 0;
        }
        else run++;
    }
}
```

$$VLC = n \times WriteTime + \beta \quad (3)$$
$$(\because n = \text{바이트저장함수의호출횟수})$$

$$WriteTime = \frac{inst}{CPUClock} \quad (4)$$

가변길이부호화 과정에 소요되는 작업 시간은 식 (3)과 같이 표현할 수 있다. n은 각 블록에 발생하는 0이 아닌 값을 가지는 AC 계수가 연속적으로 발생하는 횟수를 나타내며 WriteTime은 putAC 함수로 1 바이트의 데이타를 파일에 쓰는데 걸리는 시간을 의미한다. β는 연속적으로 0 값을 가지는 AC 계수의 출현 빈도를 세는 등의 기타 작업에 소요되는 시간이다. 이 때 WriteTime에 해당하는 작업 시간을 구하기 위해 putAC 함수의 어셈블리어 코드를 분석하였다. 프로그램을 구성하는 어셈블리어 명령의 개수는 프로세서의 종류에 따라 다를 수 있으며 본 연구에서는 IA-32의 명령어 체계를 따르는 인텔과 AMD 프로세서를 고려하였다. 읽어 들인 AC 계수의 수가 정해진 범위를 벗어나지 않고 기타 오류 없이 동작한다고 가정했을 때 putAC 함수를 호출할 경우 총 55개의 명령이 실행되는 것을 확인하였다. CPU의 한 클럭 당 하나의 명령을 수행한다면 putAC 함수를 수행하는데 걸리는 시간은 식 (4)와 같다. 또한 원본 비트율과 변환할 비트율의 차를 통해 변환한 비트율의 높고 낮음에 따라 0이 아닌 값을 가지는 AC 계수가 연속적으로 발생하는 빈도 n의 상대적인 값을 구할 수 있다. 이와 같은 내용으로 식 (3)과 식 (4)를 정리하여 식 (5)로 표현할 수 있다. 식 (5)의 계산 결과는 비트율에 따라 상대적인 것으로 변환할 비트율에 따라 식 (5)의 계산 결과만큼 적은 트랜스코딩 작업 시간을 소모하게 된다. 식 (6)은 식 (5)와 동일한 내용으로 상수항과 변수 항에 따라 정리한 것이다. 식 (6)에 의하면 상대적인 트랜스코딩 시간 Tr_R은 원본 비트율과 변환한 비트율의 차에 비례하며, 변환할 비트율이 클수록 그 값이 작아져 보다 높은 비트율로 변환할수록 더 많은 트랜스코딩 작업 시간을 소요함을 알 수 있다.

$$Tr_R = \frac{(SourceBitrate - TargetBitrate) \times playtime}{UnitOfWrite} \times \frac{inst}{CPUClock} \quad (5)$$

$$Tr_R = \left(\frac{inst \times playtime}{UnitOfWrite \times CPUClock} \right) \times (SourceBitrate - TargetBitrate) \quad (6)$$

식 (1)에서 부터 유도한 식 (6)에 의하여 임의의 트랜스코딩 서버에서 임의의 영화를 트랜스코딩하는데 소요되는 상대적인 시간을 예측할 수 있었다. 그러나 이러한 상대적인 시간을 이용해서는 실제 분산 트랜스코딩 서비스 환경에서의 트랜스코딩 서버 간 부하 균형을 맞추기 어렵다. 따라서 상대적인 트랜스코딩 시간을 절대적

인 트랜스코딩 시간으로 변환하는 과정이 필요하다. 식 (7)에서 식 (11)까지의 식은 상대적인 트랜스코딩 시간을 이용하여 절대적인 트랜스코딩 시간을 예측하기 위한 식을 보여준다. 앞서 언급한 바와 같이 트랜스코딩 목적 비트율에 따라 트랜스코딩 시간이 비례하기 때문에, 상대적인 트랜스코딩 시간과 원본 비트율로 트랜스코딩했을 때의 시간의 비와 트랜스코딩 목적 비트율 값들의 비를 이용하여 식 (7)을 유도할 수 있다. 식 (7)을 전개하여 풀어보면 최종적으로 식 (11)을 이용하여 절대적인 트랜스코딩 시간을 예측할 수 있다.

$$SourceBitrate : DiffBitrate = STT : (x \times Tr_R) \quad (7)$$

$$DiffBitrate \times STT = x \times Tr_R \times SourceBitrate \quad (8)$$

$$x = \frac{DiffBitrate \times STT}{Tr_R \times SourceBitrate} \quad (9)$$

$$x' = x \times \frac{SourceBitrate}{DiffBitrate} \quad (10)$$

$$Tr_T = x' \times Tr_R \quad (11)$$

4. 실험 결과 및 분석

4.1 실험 환경

앞 장에서는 분산 트랜스코딩 서비스 환경에서 트랜스코딩 서버간의 부하 균형을 위해 트랜스코딩 서버에서의 트랜스코딩 시간을 예측하였다. 예측된 트랜스코딩 시간이 타당함을 보이기 위해 분산 트랜스코딩 시스템을 구현하였다. 실험을 위한 분산 트랜스코딩 시스템의 서버는 사용자의 서비스 요청을 받아들이고, 이를 각 트랜스코딩 서버로 서비스 요청을 전달하는 하나의 부하 분배 서버와 트랜스코딩 작업을 수행하는 다수의 트랜스코딩 서버로 구성하였다. 실험에 사용한 트랜스코딩 서버의 사양은 표 3과 같다. 두 종류의 트랜스코딩 서버는 각각 2.2GHz과 2.0GHz의 클럭으로 동작하는 AMD 계열의 CPU를 장착한 서버를 사용하였다.

미디어 데이타의 비트율을 변환하는 트랜스코딩 작업을 수행하는 트랜스코더는 오픈 소스 프로젝트로 개발되고 있는 ffmpeg 0.4.9[14] 버전을 사용하였다. 표 4는 실험에 사용한 MPEG-2 미디어 데이타를 보여준다.

4.2 결과 및 분석

표 5는 각 서버에서 표 4의 세 가지 미디어 데이타를 트랜스코딩하였을 경우의 상대적인 트랜스코딩 시간을 식 (6)을 이용하여 예측한 값을 표로 나타낸 것이다. 각 미디어 데이타의 해상도에 따라 최소한으로 가질 수 있는 비트율이 한계가 있기 때문에 일정 수준 이하의 목적 트랜스코딩 비트율로 트랜스코딩을 하여도 실제 트랜스코딩 된 데이타는 그 일정 수준 이하로 변경되지 않는다. 따라서 각 수준별 데이타를 이용하여 각 목적 트랜스코딩 비트율로 트랜스코딩을 하였다. 표 5에 따르면 서버 1에서 미디어 데이타의 비트율을 100Kbps로 트랜스코딩할 경우, 원본 비트율인 1362Kbps로 변환할 때 소요되는 트랜스코딩 작업 시간보다 상대적으로 5.446초 적은 시간이 소요된다는 것을 알 수 있다.

이렇게 예측한 상대 트랜스코딩 시간 값을 이용하여 식 (11)에 대입하면 각 서버에서 미디어 데이타를 트랜스코딩할 때 소요되는 시간을 예측할 수 있다. 각 서버에서 예측한 시간과 실제 트랜스코딩에 소요된 시간을 측정하여 그림 8, 9, 10에서 보여 준다. 예측한 값은 해당 서버에서 순수하게 트랜스코딩 작업에 소요되는 시간만을 예측한 것이지만, 실제 트랜스코딩 작업 시간을 측정할 경우 운영체제에서 소요되는 시간과 다른 프로

표 3 트랜스코딩 서버 환경

구분	CPU	메모리	운영체제
트랜스코딩서버1	AMD Athlon MP Throughbred 2200+ 1.80GHz	1GB	RedHat Linux 7.4 (Kernel 2.4.18)
트랜스코딩서버2	AMD Opteron 248 2.20GHz	1GB	Asianux 2.0 (Kernel 2.6.9)

표 4 실험에 사용한 MPEG-2 미디어 데이타

구분		비트율	재생시간	프레임율	해상도
반지의 제왕 - 두개의 탑	CIF급	1362Kbps	1448sec	24fps	656×320
	QCIF급	769Kbps	1448sec	24fps	328×160
	SQCIF급	468Kbps	1448sec	24fps	164×80

표 5 상대 트랜스코딩 작업 시간 예측 결과

QCIF급	비트율	100Kbps	200Kbps	300Kbps	400Kbps	500Kbps
	서버 1	3.783679초	3.218107초	2.652535초	2.086962초	1.52139초
	서버 2	3.096453초	2.633605초	2.170757초	1.707909초	1.245061초
SQCIF급	비트율	600Kbps	700Kbps	800Kbps	900Kbps	1000Kbps
	서버 1	4.309662초	3.744089초	3.178517초	2.612944초	2.047372초
	서버 2	3.526902초	3.064054초	2.601206초	2.138358초	1.67551초

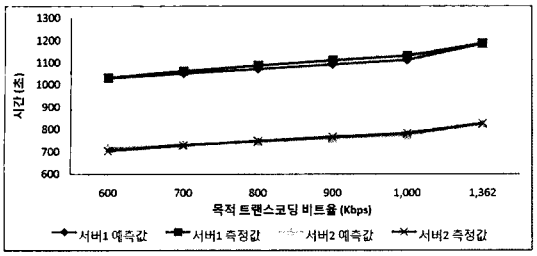

그림 8 CIF 급 영화를 사용한 트랜스코딩 시간

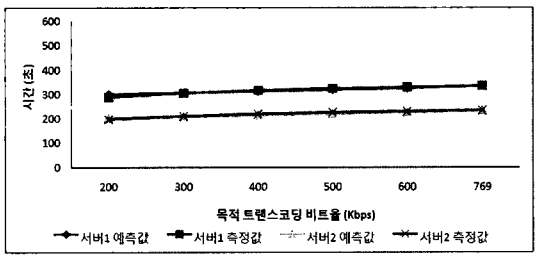

그림 9 QCIF 급 영화를 사용한 트랜스코딩 시간

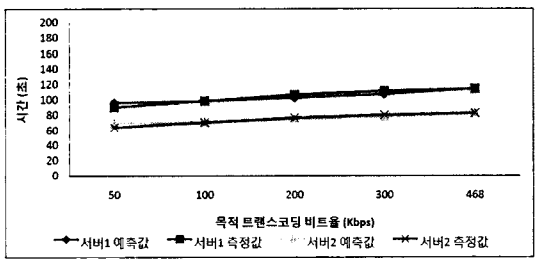

그림 10 SQCIF 급 영화를 사용한 트랜스코딩 시간

세스와의 문맥 교환에 필요한 시간 등 추가적인 시간이 측정될 수 있다. 그러한 특성을 감안하지 않더라도 그림 8, 9, 10에서의 예측 시간과 실측 시간은 상당히 유사함을 확인 할 수 있다. 실험을 통해 측정한 시간은 매 측정 마다 오차를 가질 수 있음을 감안한다면, 예측된 트랜스코딩 작업 시간은 실제 트랜스코딩 시간과 유사하다고 할 수 있다. 이러한 오차를 줄이기 위해 20회 이상 트랜스코딩 작업 시간을 측정하고 평균을 내어 그림 8, 9, 10에 나타내었다. 따라서 식 (6)을 이용하여 상대적인 트랜스코딩 시간을 예측하고, 이를 식 (11)에 대입함으로써 트랜스코딩 시간을 예측할 수 있음을 확인하였다. 이렇게 예측된 트랜스코딩 시간은 분산 트랜스코딩 시스템 환경에서 각 서버별로 정확한 부하를 예측하여 서버 간 부하 균형을 이룰 수 있는 부하 분배 알고리즘에 적용 될 수 있다.

5. 결론 및 향후 연구

무선 네트워크 환경 및 다양한 수준의 시스템 자원을 가지는 이동 단말 사용자들에게 적절한 QoS를 지원하는 스트리밍 미디어 서비스를 제공하기 위해 원본 미디어 데이타를 각 단말 환경에 맞추어 변환 제공하는 트랜스코딩 서비스가 활발하게 연구되고 있다. 트랜스코딩은 큰 시스템 자원을 필요로 하기 때문에 다수의 트랜스코딩 서버로 구성된 분산 트랜스코딩 시스템을 필요로 한다. 분산 트랜스코딩 시스템 환경에서는 스트리밍 미디어를 각 서버에서 실시간으로 트랜스코딩하여 서비스해야 하므로 각 서버간의 균형적인 부하 분배가 중요하다.

본 논문에서는 분산 트랜스코딩 서버 환경에서 균형적인 부하 분배를 위한 트랜스코딩 부하 예측 기법을 제안하였다. MPEG-2 미디어 데이타의 트랜스코딩 세부 과정을 분석하였고, 이를 바탕으로 비트율을 변경하는 트랜스코딩 과정에서는 VLC 부분이 트랜스코딩 작업 시간에 가장 큰 영향을 주는 것을 확인하였다. VLC 부분에서 소요되는 작업량을 분석하여 실제 트랜스코딩 작업에 미치는 영향을 수식을 통하여 보여주고, 이를 이용하여 비트율에 따라 소요되는 상대적인 트랜스코딩 시간을 예측하였다. 예측한 상대적인 트랜스코딩 시간과 원본 비트율로 트랜스코딩할 때의 시간을 이용하여 비례식을 만들어 실제 트랜스코딩에 소요되는 시간을 예측하는 식을 수립하였다. 이를 통해 각 서버의 CPU와 MPEG-2 미디어 데이타의 정보를 대입하고 트랜스코딩 시간에 소요되는 시간을 예측하였다.

제안한 기법을 이용하여 예측한 트랜스코딩 시간을 실제 분산 트랜스코딩 시스템을 구현하여 각 트랜스코딩 서버에서 측정한 트랜스코딩 시간과 비교 분석 하였다. 그 결과 예측한 트랜스코딩 시간과 측정한 트랜스코딩 시간이 일치함을 확인하였다. 향후에는 본 논문에서 제안한 기법을 확장하여 MPEG-4와 H.264등의 다양한 영상 압축 기법에 대한 트랜스코딩 부하를 예측하는 기법을 연구한다. 이를 통해 대규모 분산 트랜스코딩 시스템에서의 효과적인 부하 분배 알고리즘에 대해 연구하고자 한다.

참 고 문 헌

[1] Dinkar Sitaram, Asit Dan, "Multimedia Servers: Applications, Environments, and Design," Morgan Kaufmann Publishers, 2000.

[2] W.C. Feng and M. Lie, "Critical Bandwidth Allocation Techniques for Stored Video Delivery Across Best-Effort Networks," The 20th International Conference on Distributed Computing Systems, pp.201-207, April 2000.

[3] D.H.C. Du and Y. J. Lee, "Scalable Server and

Storage Architectures for Video Streaming," IEEE International Conference on Multimedia Computing and Systems, pp.191-206, June 1999.
[4] Dongmahn Seo, Joahyoung Lee, Yoon Kim, Changyeol Choi, Hwangkyu Choi, Inbum Jung, "Load Distribution Strategies in Cluster-based Transcoding Servers for Mobile Clients," Lecture Notes in Computer Science, Vol 3983, pp. 1156-1165, May 2006.
[5] Dongmahn Seo, Joahyoung Lee, Yoon Kim, Changyeol Choi, Manbae Kim, Inbum Jung, "Resource Consumption-Aware QoS in Cluster-based VOD Servers," Journal of Systems Architecture: the EUROMICRO Journal, Volume 53, Issue 1, pp. 39-52, Jan. 2007.
[6] H.Bhradvaj, A. Joshi and S. Auephanwiriyakul. "An active transcoding proxy to support mobile web access," In Proceedings of International Conference on Reliable Distrubuted System, pp 118-123, 1998.
[7] Vetro. A.; Sun, H., "Media Conversions to Support Mobile Users," IEEE Canadian Conference on Electrical and Computer Engineering (CCECE), pp. 607-612, May. 2001.
[8] 이호석, 김준기, "알기 쉬운 MPEG-2 소스코드 해설", 홍릉과학출판사, 2001.
[9] MPEG 연구 사이트, http://www.mpeg.org
[10] MPEG 홈 페이지, http://www.chiariglione.org/mpeg/
[11] Sumit Roy, Michele Covell, John Ankcorn, and Susie Wee, "A System Architecture for Managing Mobile Streaming Media Services," Takeshi Yoshimura Streaming Media Systems Group, Hewlett-Packard Laboratories, Palo Alto, CA 94304.
[12] 이성용, "리눅스 기반 모바일 미디어 스트리밍 시스템의 설계 및 구현," 강원대학교 공학석사 학위 논문, 2005년.
[13] Anthony Vetro, Charilaos Christopoulos, and Huifang Sun, "Video Transcoding Architectures and Techniques: An Overview," IEEE Signal Processing Magazine, Vol. 20, Issue 2, pp. 18-29, Mar. 2003.
[14] ffmpeg 개발 사이트, http://ffmpeg.sourceforge.net

서 동 만

2002년 2월 강원대학교 컴퓨터학과 학사 2004년 2월 강원대학교 컴퓨터정보통신공학과 석사. 2004년 3월~현재 강원대학교 컴퓨터정보통신공학과 박사 수료, 연구과정생. 관심분야는 병렬처리, 멀티미디어 시스템, 운영체제, 센서네트워크

허 난 숙

2006년 2월 강원대학교 전기전자정보통신공학부 컴퓨터전공 학사. 2008년 8월 강원대학교 컴퓨터정보통신공학과 석사 졸업 예정. 2008년 2월~현재 NHN서비스(주) 고객마케팅개발팀. 관심분야는 멀티미디어 시스템, 미디어 트랜스코딩

김 종 우

2002년 3월~현재 강원대학교 컴퓨터정보통신공학과 학부과정. 관심분야는 멀티미디어 시스템, 센서 네트워크

정 인 범

1985년 고려대학교 전자공학과 졸업(학사). 1985년~1995년 (주) 삼성전자 컴퓨터 시스템사업부 선임 연구원. 1992년~1994년 한국과학기술원 정보및통신공학과 졸업(컴퓨터공학 석사). 1995년 2000년 8월 한국과학기술원 전산학과 졸업(박사). 2001년~현재 강원대학교 전기전자정보통신공학부 컴퓨터전공 교수. 관심분야는 운영체제