트랜스코딩 서버 간 부하 분산을 위한 트랜스코딩 부하 예측

김종우, 허난숙, 서동만, 정인범*
강원대학교 컴퓨터정보통신공학전공
e-mail:jw_kim@snslab.kangwon.ac.kr

Transcoding Load Estimation for Load Distribution between Transcoding Servers

Jong-Woo Kim, Nan-Sook Heo, Dong-Mahn Seo, In-Bum Jung*
Dept of Computer Engineering, Kangwon National University

요 약

무선 네트워크의 발달로 인해 이동 사용자가 급증함에 따라 무선 네트워크 환경에서도 질 높은 미디어 스트리밍 서비스를 받기 위한 수요가 증가하고 있다. 그러나 무선 네트워크는 유선 네트워크보다 상대적으로 좁은 네트워크 대역폭을 가진다. 또한 유선 네트워크 환경의 단말기에 비해 제한된 시스템 자원을 가지므로 유선 네트워크 환경의 사용자를 위한 고화질 미디어 데이터를 그대로 스트리밍 서비스 받기에 적합하지 않다. 최근 이러한 문제를 해결하기 위한 방법으로 미디어 트랜스코딩 서비스가 활발하게 연구되고 있다. 다수의 트랜스코딩 서버로 구성된 분산 트랜스코딩 시스템에서 특정 서버로 트랜스코딩 작업이 집중될 경우 사용자에게 질 높은 수준의 미디어 스트리밍 서비스를 제공하기 어렵다. 때문에 각 트랜스코딩 서버간의 균형적인 부하 분산이 중요한 문제가 되며, 처리되는 미디어 데이터의 특성을 부하 분산에 반영할 필요가 있다. 본 논문에서는 MPEG-2 미디어 데이터의 트랜스코딩 과정을 분석함으로써 비트율 변경에 따른 트랜스코딩 작업 시간 예측 기법을 제안한다. 제안된 기법은 트랜스코딩 서버 정보와 영화 정보, 목적 트랜스코딩 비트율을 이용하여 예상 트랜스코딩 시간을 예측한다.

1. 서론

무선 네트워크의 발달로 인해 이동 사용자가 급증함에 따라 무선 네트워크 환경에서도 질 높은 미디어 스트리밍 서비스를 받기 위한 수요가 증가하고 있다[1, 2, 3, 4, 5]. 이러한 수요를 충족시키기 위해 많은 연구가 진행되고 있다[4, 6, 7]. 본 논문에서는 다수의 트랜스코딩 서버로 구성된 분산 트랜스코딩 서버에서 효율적으로 부하를 분산하기 위한 트랜스코딩 부하 예측 기법을 제안한다. 제안하는 기법은 미디어 데이터의 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.

2. 관련연구

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. MPEG에서 사용하는 압축 기술로 DCT(Discrete Cosine Transform)와 양자화(Quantization), 가변 길이 부호화(Variable Length Coding)가 있다[8, 9].

- 본 연구는 산업자원부와 한국산업기술재단의 지역혁신인력 양성사업으로 수행된 연구 결과임.

*교신저자

일반적인 트랜스코딩 시스템에서 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. 기존의 트랜스코딩 시스템을 구축하는 방법들로 소스기반 정적 인코딩 시스템과 트랜스코딩 서버 시스템이 있다[4, 10]. 정적 트랜스코딩 서버 방식의 문제점인 특정 서버에 부하가 집중되는 부하 불균형을 피하기 위하여 분산 서버를 두고 트랜스코딩 서버들의 부하 상태를 파악하여 트랜스코딩 요구들을 처리하는 부하 분산 트랜스코딩 시스템 방식이 있다. 이 방식에서 분산서버는 트랜스코딩 서버들과 연동하여 정해진 부하 분산 정책에 따라서 트랜스코딩 서버를 선택하게 된다. 전통적으로 클러스터 시스템에서는 부하 분산 정책으로, 라운드로빈, 가중치기반 라운드로빈, 가중치기반 최소 접속 알고리즘 등이 사용되고 있다[4, 6, 7].

3. 트랜스코딩 작업 시간 예측

트랜스코딩은 원본 미디어 데이터를 입력받아 목적하는 형태로 변환하는 작업을 수행한다. 가장 단순한 트랜스코딩 방법은 원본 미디어 데이터를 픽셀 단위까지 완전하게 디코딩 한 후 사용될 목적에 적합하도록 비트율, 프레임율, 해상도 등을 변경한 후 다시 인코딩한다. 이러한 트랜스코딩 방법을 픽셀 도메인(Pixel-domain) 트랜스코딩이

<표 1> 논문에서 사용된 기호 및 의미.

기 호	의 미	단위
Tr_T	트랜스코딩 작업 수행에 소요되는 시간	sec
Tr_R	변환 비트율에 따라 상대적으로 소요되는 트랜스코딩 작업 수행 시간	sec
SourceBitrate	미디어 데이터의 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 데이터를 메모리에 쓰는데 소요되는 시간	sec
CPUClock	트랜스코딩 서버의 CPU Clock	Hz
playtime	미디어 데이터의 재생시간	sec
INST	putAC 함수를 호출할 경우 실행되는 명령어의 개수	개
UnitOfWrite	putAC 함수에서 데이터를 파일에 쓰는 최소 단위	bit
STT	원본 비트율과 동일하게 트랜스코딩 수행할 때 소요되는 시간	sec

라 한다. 픽셀 도메인 트랜스코딩은 높은 CPU 처리량을 요구하며, 처리해야 하는 데이터의 양이 크기 때문에 많은 메모리 용량을 필요로 한다. CPU와 메모리 등의 시스템 자원의 활용 면에서 보다 효율적인 트랜스코딩 방법으로 미디어 데이터를 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.

트랜스코딩 작업 시간은 트랜스코딩 각 과정의 수행 시간을 합한 것으로, 표 1의 기호를 이용하여 수식으로 정리하면 식 1과 같이 표현할 수 있다. 가변길이 복호화(VLD)와 역양자화(iQ), 역DCT(iDCT) 과정을 거쳐 미디어 데이터를 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 재 인코딩하는 일반적인 트랜스코딩 과정의 작업시간이다.

$$Tr_T = VLD + iQ + iDCT + MC + Sampling + ME/MC + DCT + Q + VLC$$

식 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 시간에 영향을 주지 않는다. 따라서 비트율만 변경하는 트랜스코딩의 작업 시간은 비트율 변경에 직접적으로 영향을 주는 가변길이부호화와 그 외 과정의 작업시간의 합으로 식 2와 같이 표현할 수 있다.

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

식 2.

가변길이부호화(VLC)는 양자화 결과 값을 읽어들여 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. 0인 양자화 결과 값이 연속적으로 나타나는 빈도를 셀 때 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, 11]. 8×8 크기의 블록 단위로 양자화가 수행되기 때문에 한 양자화 결과 값은 블록당 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 데이터를 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.

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

식 3.

$$WriteTime = \frac{inst}{CPUClock}$$

식 4.

가변길이부호화 과정에 소요되는 작업 시간은 식 3과

<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++;
    }
}
```

같이 표현할 수 있다. n은 각 블록에 발생하는 0이 아닌 값을 가지는 AC 계수가 연속적으로 발생하는 횟수를 나타내며 WriteTime은 putAC 함수로 1 바이트의 데이터를 파일에 쓰는데 걸리는 시간을 의미한다. β는 연속적으로 0 값을 가지는 AC 계수의 출현빈도를 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 4와 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 3과 식 4를 정리하여 식 5로 표현할 수 있다. 식 5의 계산 결과는 비트율에 따라 상대적인 것으로 변환할 비트율에 따라 식 5의 계산 결과만큼 적은 트랜스코딩 작업 시간을 소모하게 된다. 식 6은 식 5와 동일한 내용으로 상수항과 변수항에 따라 정리한 것이다. 식 6에 의하면 상대적인 트랜스코딩 시간 TrR은 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.

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

식 5.

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

식 6.

식 1에서 부터 유도한 식 6에 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 절대적인 트랜스코딩 시간으로 변환하는 과정이 필요하다. 식 7에서 식 11일까지의 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 트랜스코딩 했을 때의 시간의 비와 트랜스코딩 목적 비트율 값들의 비를 이용하여 식 7을 유도할 수 있다. 식 7을 전개하여 풀어보면 최종적으로 식 11을 이용하여 절대적인 트랜스코딩 시간을 예측할 수 있다.

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

식 7.

$$DiffBitrate \times STT = x \times Tr_R \times SourceBitrate$$

식 8.

$$x = \frac{DiffBitrate \times STT}{Tr_R \times SourceBitrate}$$

식 9.

$$x' = x \times \frac{SourceBitrate}{DiffBitrate}$$

식 10.

$$Tr_T = x' \times Tr_R$$

식 11.

4. 실험 결과 및 분석

예측된 트랜스코딩 시간이 타당함을 보이기 위해 분산 트랜스코딩 시스템을 구현하였다. 서버는 사용자의 서비스 요청을 받아들이고, 이를 각 트랜스코딩 서버로 서비스 요청을 전달하는 하나의 부하 분산 서버와 트랜스코딩 작업을 수행하는 다수의 트랜스코딩 서버로 구성하였다. 실험에 사용한 트랜스코딩 서버 각각 2.2GHz과 2.0GHz, 1.80GHz의 클럭으로 동작하는 AMD 계열의 CPU를 장착한 서버를 사용하였다. 미디어 데이터의 비트율을 변환하는 트랜스코딩 작업을 수행하는 트랜스코더는 오픈 소스 프로젝트로 개발되고 있는 ffmpeg 0.4.9[12] 버전을 사용하였다. 실험에 사용한 MPEG-2 미디어 데이터는 반지의 제왕-두개의 탑(비트율:1362Kbps, 재생시간:1448sec, 프레임율:24fps, 해상도:656×320)을 사용하였다.

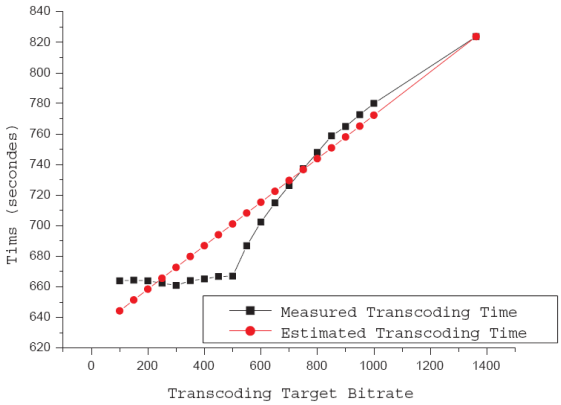
표 3은 각 서버에서 MPEG-2 미디어 데이터를 100Kbps부터 1000Kbps까지의 비트율로 트랜스코딩 하였을 경우의 상대적인 트랜스코딩 시간을 식 6을 이용하여 예측한 값을 표로 나타낸 것이다. 표 3에 따르면 서버 1에서 미디어 데이터의 비트율을 100Kbps로 트랜스코딩 할 경우, 원본 비트율인 1362Kbps로 변환할 때 소요되는 트랜스코딩 작업 시간보다 상대적으로 5.446초 적은 시간이 소요된다는 것을 알 수 있다.

이렇게 예측한 상대 트랜스코딩 시간 값을 이용하여 식 11에 대입하면 각 서버에서 미디어 데이터를 트랜스코딩 할 때 소요되는 시간을 예측할 수 있다. 각 서버에서 예측한 시간과 실제 트랜스코딩에 소요된 시간을 측정하여 그림 1, 2, 3에서 보여 준다. 예측한 값은 해당 서버에서 순수하게 트랜스코딩 작업에 소요되는 시간만을 예측한 것이지만, 실제 트랜스코딩 작업 시간을 측정할 경우 운영체제에서 소요되는 시간과 다른 프로세스와의 문맥 교환에 필요한 시간 등 추가적인 시간이 측정될 수 있다. 그러한 특성을 감안하지 않더라고 그림 1, 2, 3에서의 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. 따라서 식 6을 이용하여 상대적인 트랜스코딩 시간을 예측하고, 이를 식 11에

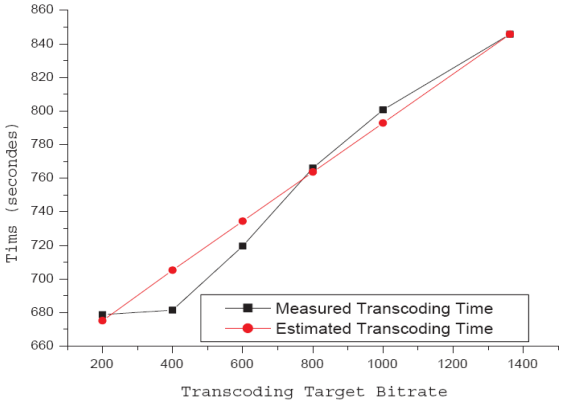
<표 3> 상대적인 트랜스코딩 작업 시간 예측 결과.

비트율	200Kbps	400Kbps	600Kbps	800Kbps	1000Kbps
서버 1	5.01447 sec	4.15139 sec	3.28832 sec	2.42524 sec	1.56217 sec
서버 2	5.43 sec	4.473 sec	3.543 sec	2.613 sec	1.683 sec
서버 3	6.483 sec	5.367 sec	4.251 sec	3.135 sec	2.019 sec

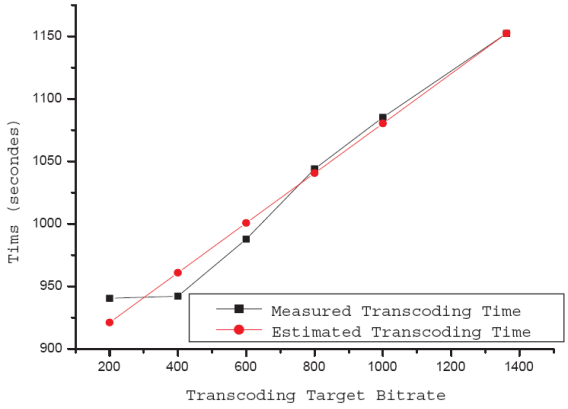
(그림 1) 서버 1에서의 트랜스코딩 시간.



(그림 2) 서버 2에서의 트랜스코딩 시간.



(그림 3) 서버 3에서의 트랜스코딩 시간.



대입함으로써 트랜스코딩 시간을 예측할 수 있음을 확인하였다. 이렇게 예측된 트랜스코딩 시간은 분산 트랜스코딩 시스템 환경에서 각 서버별로 정확한 부하를 예측하여 서버 간 부하 균형을 이를 수 있는 부하 분산 알고리즘에 적용 될 수 있다.

5. 결론 및 향후 계획

본 논문에서는 분산 트랜스코딩 서버 환경에서 균형적인 부하 분산을 위한 트랜스코딩 부하 예측 기법을 제안하였다. 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 분산 알고리즘에 대해 연구하고자 한다.

참고문헌

[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] 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.

[11] MPEG 홈 페이지, http://www.chiariglione.org/mpeg/

[12] ffmpeg 개발 사이트, http://ffmpeg.sourceforge.net