



(19) 대한민국특허청(KR)
(12) 등록특허공보(B1)

(45) 공고일자 2008년11월19일
(11) 등록번호 10-0869513
(24) 등록일자 2008년11월13일

(51) Int. Cl.

H04N 7/01 (2006.01) H04N 7/24 (2006.01)

H04N 7/26 (2006.01)

(21) 출원번호 10-2007-0123185

(22) 출원일자 2007년11월30일

심사청구일자 2007년11월30일

(56) 선행기술조사문헌

JP19506368 T*

KR1020050036998 A*

*는 심사관에 의하여 인용된 문헌

(73) 특허권자

인하대학교 산학협력단

인천 남구 용현동 253 인하대학교

(72) 발명자

송민석

서울 서초구 방배3동 래미안 아트힐 107동 402호

심정섭

인천 남동구 간석2동 금호어울림아파트 110동 1101호

(뒷면에 계속)

(74) 대리인

김건우

전체 청구항 수 : 총 2 항

심사관 : 구대성

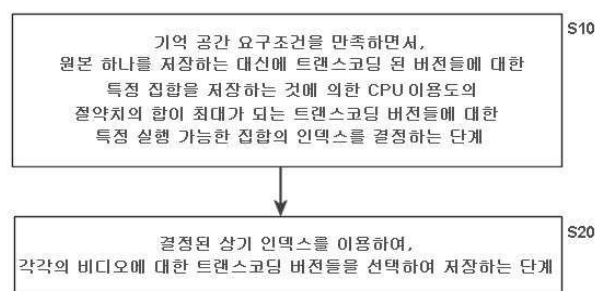
(54) 트랜스코딩 서버의 기억 공간을 관리하는 방법

(57) 요약

본 발명은 트랜스코딩 서버의 기억 공간을 관리하는 방법에 관한 것으로서, 보다 구체적으로는, 적어도 하나 이상의 비디오에 대하여 트랜스코딩 버전을 관리하는 트랜스코딩 서버에 있어서, 상기 적어도 하나 이상의 비디오 각각(V_i)에 대하여, 각각의 비디오의 선택된 트랜스코딩 버전들로부터 요구되는 전체 기억 공간의 양(STR_{i,SE_i})의 합이 상기 트랜스코딩 서버에서 사용가능한 전체 기억 공간의 양(TS)보다 작거나 같다는 조건을 만족하면서, 원본 하나를 저장하는 대신에 트랜스코딩 된 버전들에 대한 특정 집합을 저장하는 것에 의한 CPU 이용도의 절약치(SV_{i,SE_i})의 합이 최대가 되는 트랜스코딩 버전들에 대한 특정 실행 가능한 집합(FS_{i,SE_i})의 인덱스(SE_i)를 결정하는 단계; 및 결정된 상기 인덱스(SE_i)를 이용하여 상기 적어도 하나 이상의 비디오 각각(V_i)에 대한 트랜스코딩 버전들을 선택하여 저장하는 단계를 포함하는, 트랜스코딩 서버의 기억 공간 관리 방법에 관한 것이다.

본 발명에서 제안하고 있는 트랜스코딩 서버의 기억 공간 관리 방법에 따르면, 트랜스코딩과 관련하여 발생하는 기억 공간과 CPU 자원 사이의 트레이드-오프를 효과적으로 밸런싱 할 수 있으며, 그 결과 주어진 기억 공간을 이용하여 CPU 이용도에 있어서의 최대 절약치를 얻을 수 있다.

대표도 - 도1



(72) 발명자

고재두

경북 울진군 원남면 매화2리 758번지

이범선

인천 남구 학익동 437-14 솔로타운 B-204

특허청구의 범위

청구항 1

적어도 하나 이상의 비디오에 대하여 트랜스코딩(transcoding) 버전을 관리하는 트랜스코딩 서버에 있어서,
상기 적어도 하나 이상의 비디오 각각(V_i)에 대하여, 각각의 비디오의 선택된 트랜스코딩 버전들로부터 요구되는 전체 기억 공간의 양(STR_{i,SE_i})의 합이 상기 트랜스코딩 서버에서 사용가능한 전체 기억 공간의 양(TS)보다 작거나 같다는 조건을 만족하면서, 원본 하나를 저장하는 대신에 트랜스코딩 된 버전들에 대한 특정 집합을 저장하는 것에 의한 CPU 이용도의 절약치(SV_{i,SE_i})의 합이 최대가 되는 트랜스코딩 버전들에 대한 특정 실행 가능한 집합(FS_{i,SE_i})의 인덱스(SE_i)를 결정하는 단계; 및

결정된 상기 인덱스(SE_i)를 이용하여 상기 적어도 하나 이상의 비디오 각각(V_i)에 대한 트랜스코딩 버전들을 선택하여 저장하는 단계를 포함하며,

트랜스코딩 버전들에 대한 특정 실행 가능한 집합(FS_{i,SE_i})의 인덱스(SE_i)를 결정하는 상기 단계는, 다음 수학적 식을 이용하여 NV개의 비디오 대하여 사용가능한 기억 공간이 TS인 경우의 CPU 이용도에서의 최대 절약치를 계산함으로써 상기 인덱스를 결정하는, 트랜스코딩 서버의 기억 공간을 관리하는 방법.

$$OPT_{v,w} = \max_{1 \leq j \leq NE} \{ \max \{ OPT_{v-1,w}, OPT_{v-1,w-STR_{v,j}} + SV_{v,j} \} \}$$

여기서, $OPT_{v,w}$ ($1 \leq v \leq NV$, $1 \leq w \leq TS$)는 비디오 $V_1 \dots V_v$ 에 대하여 할당된 디스크 기억 공간이 w인 경우의 최대 CPU 절약치를 나타내고, $STR_{v,j}$ 는 비디오 V_v 의 저장된 버전들의 모든 실행 가능한 집합들을 원소로 하는 집합 F_v 의 j 번째 원소 $FS_{v,j}$ 내의 버전들에 대한 전체 기억 공간 요구조건을 나타내며, $SV_{v,j}$ 는 원본 집합 $FS_{v,1}$ 내의 버전들을 저장하는 대신에 집합 $FS_{v,j}$ 내의 버전들을 저장하는 것에 의한 CPU 이용도의 절약치를 나타낸다.

청구항 2

삭제

청구항 3

삭제

청구항 4

제1항에 있어서,

상기 트랜스코딩 서버는 프락시 서버인, 트랜스코딩 서버의 기억 공간을 관리하는 방법.

명세서

발명의 상세한 설명

기술 분야

<1> 본 발명은 트랜스코딩 서버의 기억 공간을 관리하는 방법에 관한 것으로서, 보다 구체적으로는, 트랜스코딩 서버에 저장되는 각각의 비디오에 대하여 원본 하나를 저장하는 대신에 트랜스코딩 된 버전들에 대한 특정 집합을 저장하는 것에 의한 CPU 이용도의 절약치의 합이 최대가 되는 트랜스코딩 된 버전들에 대한 특정 집합을 결정되, 각각의 비디오의 선택된 트랜스코딩 버전들로부터 요구되는 전체 기억 공간의 양의 합이 상기 트랜스코딩 서버에서 사용가능한 전체 기억 공간의 양보다 작거나 같다는 조건을 만족하도록 함으로써, 주어진 기억 공간을 이용하여 CPU 이용도에 있어서의 최대 절약치를 얻을 수 있는 트랜스코딩 서버의 기억 공간 관리 방법에 관한 것이다.

배경 기술

- <2> 통신 및 멀티미디어 기술에서의 최근의 발전으로 인하여, 사용자는 PDA(Personal Digital Assistants), 핸드폰, 노트북 컴퓨터와 같은 이종 장치들을 이용하여 인터넷을 통해 비디오 서비스에 접근할 수 있게 되었다. 이들 장치들은 서로 다른 처리 용량, 에너지 수지(energy budget), 디스플레이 사이즈 및 네트워크 접속성(network connectivities)을 갖는다. 사용자들은 보통, 해상도, 프레임 레이트, 디스플레이 사이즈와 같은 서비스 품질(Quality of Service; QoS) 파라미터들을 가지고 자신의 필요에 부합하는 장치들을 선택한다. 예를 들어, 고속 네트워크 사용자는 무선 네트워크 사용자보다 훨씬 고품질의 비디오 서비스를 기대할 것이다.
- <3> 이와 같은 이중성을 지원하기 위하여, 비디오 콘텐츠는 사용자들의 요구를 만족시킬 수 있도록 적합한 형태로 변환될 필요가 있다. 이와 같은 변환 과정을 트랜스코딩(transcoding)이라고 지칭한다. 트랜스코딩의 전형적인 예에는, 해상도의 크기를 줄임으로써 비디오 스트림의 비트-레이트를 감소시키는 것이 포함된다. 트랜스코딩은 서버, 프록시(proxy) 또는 고객 컴퓨터(client)에서 발생할 수 있다. 고객-측에서의 트랜스코딩은 요구되는 연산 전력(computational power) 때문에 사실상 어렵다. 또한, 프록시-기반 트랜스코딩(proxy-based transcoding)에는 발행인 제어(publisher control) 및 저작권(copyright)의 문제점이 있기 때문에, 트랜스코딩은 서버에서 수행되는 것으로 가정되지만, 프록시는 최종-사용자에게 서버로서 작용하기 때문에, 트랜스코딩은 프록시에서 동작하도록 쉽게 변형될 수 있다.
- <4> 기존의 트랜스코딩은 동적 또는 정적일 수 있다. 동적 트랜스코딩에서는, 최고 품질의 버전만이 저장되고, 보다 낮은 품질의 버전들은 온라인으로 추출되어야 한다. 이 방법은 트랜스코딩 동작의 연산 요구조건 때문에 CPU 자원을 빠르게 고갈시킬 수 있다. 정적 트랜스코딩에서는, 서버가 서로 다른 품질 레벨의 다수의 버전들을 생성하여 오프-라인으로 저장하는데, 이것은 많은 디스크 공간을 필요로 한다.
- <5> 이상 살펴본 바와 같이, 트랜스코딩 방법에 따라 기억 공간과 CPU 자원 사이에 트레이드-오프가 발생한다. 따라서 트랜스코딩에 의해 만들어지는 CPU 요구를 최소화시킬 수 있도록, 디스크 상에 저장될 수 있는 비디오의 버전을 선택하는 일반화된 해석 모델이 필요하다. 또한, 상기 일반화된 해석 모델을 동적 프로그래밍 기법을 이용하여 디스크 상에 저장된 버전들로부터 적합한 버전들을 선택하는 새로운 알고리즘이 제안될 필요가 있다.

발명의 내용

해결 하고자하는 과제

- <6> 본 발명은 상기와 같은 필요성의 인식에서 비롯된 것으로서, 트랜스코딩과 관련하여 발생하는 기억 공간과 CPU 자원 사이의 트레이드-오프를 효과적으로 밸런싱 할 수 있으며, 그 결과 주어진 기억 공간을 이용하여 CPU 이용도에 있어서의 최대 절약치를 얻을 수 있는 트랜스코딩 서버의 기억 공간 관리 방법을 제공하는 것을 그 목적으로 한다.

과제 해결수단

- <7> 상기한 목적을 달성하기 위한 본 발명의 특징에 따른 트랜스코딩 서버의 기억 공간을 관리하는 방법은,
- <8> 적어도 하나 이상의 비디오에 대하여 트랜스코딩(transcoding) 버전을 관리하는 트랜스코딩 서버에 있어서,
- <9> 상기 적어도 하나 이상의 비디오 각각(V_i)에 대하여, 각각의 비디오의 선택된 트랜스코딩 버전들로부터 요구되는 전체 기억 공간의 양(STR_{i,SE_i})의 합이 상기 트랜스코딩 서버에서 사용가능한 전체 기억 공간의 양(TS)보다 작거나 같다는 조건을 만족하면서, 원본 하나를 저장하는 대신에 트랜스코딩 된 버전들에 대한 특정 집합을 저장하는 것에 의한 CPU 이용도의 절약치(SV_{i,SE_i})의 합이 최대가 되는 트랜스코딩 버전들에 대한 특정 실행 가능한 집합(FS_{i,SE_i})의 인덱스(SE_i)를 결정하는 단계; 및
- <10> 결정된 상기 인덱스(SE_i)를 이용하여 상기 적어도 하나 이상의 비디오 각각(V_i)에 대한 트랜스코딩 버전들을 선택하여 저장하는 단계를 포함하는 것을 그 구성상의 특징으로 한다.
- <11> 바람직하게는, 트랜스코딩 버전들에 대한 특정 실행 가능한 집합(FS_{i,SE_i})의 인덱스(SE_i)를 결정하는 상기 단계는, 동적 프로그래밍 기법을 이용하여 구현된다.
- <12> 바람직하게는, 트랜스코딩 버전들에 대한 특정 실행 가능한 집합(FS_{i,SE_i})의 인덱스(SE_i)를 결정하는 상기 단계는, 다음 수학적 식 1을 이용하여 NV개의 비디오 대하여 사용가능한 기억 공간이 TS인 경우의 CPU 이용도에서

의 최대 절약치를 계산함으로써 상기 인덱스를 결정한다.

수학식 1

<13>

$$OPT_{v,w} = \max_{1 \leq j \leq NE} \{ \max \{ OPT_{v-1,w}, OPT_{v-1,w-STR_{v,j}} + SV_{v,j} \} \}$$

<14>

여기서, $OPT_{v,w}$ ($1 \leq v \leq NV$, $1 \leq w \leq TS$)는 비디오 $V_1 \sim V_v$ 에 대하여 할당된 디스크 기억 공간이 w 인 경우의 최대 CPU 절약치를 나타내고, $STR_{v,j}$ 는 비디오 V_v 의 저장된 버전들의 모든 실행 가능한 집합들을 원소로 하는 집합 F_v 의 j 번째 원소 $FS_{v,j}$ 내의 버전들에 대한 전체 기억 공간 요구조건을 나타내며, $SV_{v,j}$ 는 원본 집합 $FS_{v,1}$ 내의 버전들을 저장하는 대신에 집합 $FS_{v,j}$ 내의 버전들을 저장하는 것에 의한 CPU 이용도의 절약치를 나타낸다.

<15>

바람직하게는, 상기 트랜스코딩 서버는 프락시 서버일 수 있다.

효과

<16>

본 발명에서 제안하고 있는 트랜스코딩 서버의 기억 공간 관리 방법에 따르면, 트랜스코딩과 관련하여 발생하는 기억 공간과 CPU 자원 사이의 트레이드-오프를 효과적으로 밸런싱 할 수 있으며, 그 결과 주어진 기억 공간을 이용하여 CPU 이용도에 있어서의 최대 절약치를 얻을 수 있다.

발명의 실시를 위한 구체적인 내용

<17>

이하에서는 첨부된 도면들을 참조하여, 본 발명에 따른 실시예에 대하여 상세하게 설명하기로 한다.

<18>

도 1은 본 발명의 일 실시예에 따른, 트랜스코딩 서버의 기억 공간 관리 방법의 구성을 나타내는 도면이다. 도 1에 도시된 바와 같이, 본 발명의 일 실시예에 따른, 트랜스코딩 서버의 기억 공간 관리 방법은, 기억 공간 요구조건을 만족하면서 원본 하나를 저장하는 대신에 트랜스코딩 버전들에 대한 특정 실행 가능한 집합을 저장하는 것에 의한 CPU 이용도의 절약치의 합이 최대가 되는 트랜스코딩 버전들에 대한 특정 실행 가능한 집합의 인덱스를 결정하는 단계(S10)와, 단계 S10에서 결정된 인덱스를 이용하여 각각의 비디오에 대한 트랜스코딩 버전들을 선택하여 저장하는 단계(S20)를 포함한다.

<19>

단계 S10은 후술하는 바와 같이 동적 프로그래밍 기법을 이용하여 구현될 수 있으며, 상기 트랜스코딩 서버는 프락시 서버일 수 있다. 또한, 단계 S10에서는, 앞서 언급한 바 있는 다음 수학식 1을 이용하여 NV 개의 비디오에 대하여 사용가능한 기억 공간이 TS 인 경우의 CPU 이용도에서의 최대 절약치를 계산함으로써 인덱스를 결정할 수 있다.

<20>

<수학식 1>

<21>

$$OPT_{v,w} = \max_{1 \leq j \leq NE} \{ \max \{ OPT_{v-1,w}, OPT_{v-1,w-STR_{v,j}} + SV_{v,j} \} \}$$

<22>

여기서, $OPT_{v,w}$ ($1 \leq v \leq NV$, $1 \leq w \leq TS$)는 비디오 $V_1 \sim V_v$ 에 대하여 할당된 디스크 기억 공간이 w 인 경우의 최대 CPU 절약치를 나타내고, $STR_{v,j}$ 는 비디오 V_v 의 저장된 버전들의 모든 실행 가능한 집합들을 원소로 하는 집합 F_v 의 j 번째 원소 $FS_{v,j}$ 내의 버전들에 대한 전체 기억 공간 요구조건을 나타내며, $SV_{v,j}$ 는 원본 집합 $FS_{v,1}$ 내의 버전들을 저장하는 대신에 집합 $FS_{v,j}$ 내의 버전들을 저장하는 것에 의한 CPU 이용도의 절약치를 나타낸다.

<23>

1. 시스템 모델

<24>

이중 고객들의 요구를 지원하기 위하여, 서버는 원본 비디오 객체를 서로 다른 변체(variants)로 트랜스코딩 할 수 있는데, 각각의 변체를 트랜스코딩 된 버전(transcoded version)이라고 지칭한다. 보다 높은 비트-레이트의 버전은 보다 낮은 비트-레이트의 버전으로 트랜스코딩 될 수 있지만, 비트-레이트에서의 증가는 지원되지 않는다.

<25>

각각의 비디오 V_i 가 T_i 초의 재생 길이를 가지며, NR 개의 버전(원본 버전 V_i^1 과 트랜스코딩 된 버전 $V_i^2 \sim V_i^{NR}$, $i = 1, \dots, NV$)이 존재한다고 가정하자. b_i^k 가 버전 V_i^k ($k = 1, \dots, NR$)의 비트-레이트를 나타낸다고 하면, 가장 높은 비트-레이트의 버전은 V_i^1 이고, 가장 낮은 비트-레이트의 버전은 V_i^{NR} 이다. 즉, $b_i^1 > \dots > b_i^{NR}$.

- <26> 모든 비디오의 접근 확률이 미리 알려져 있다고 가정할 것이다. 고객들의 요구 패턴을 모니터링 함으로써 이들 확률을 얻을 수 있다. p_i^k 을 버전 V_i^k ($k = 1, \dots, NR$)의 접근 확률이라고 하자(여기서, $\sum_{i=1}^{NV} \sum_{k=1}^{NR} p_i^k = 1$).
- <27> 고객의 요구를 수신하면, 서버는 자신의 디스크를 검색하여 적합한 버전을 찾는다. 요구된 버전이 디스크 상에 저장되어 있으면, 해당 버전이 고객에게 직접 보내진다. 그렇지 않으면, 트랜스코딩이 요구된다. 만약 충분한 CPU 자원이 사용가능하다면, 서버는 트랜스코딩을 시작하여, 결과 스트림을 고객에게 보낸다. 이것도 아니면, 요구는 차단된다. CPU의 사용을 줄임으로써, 서버는 보다 많은 고객을 수용할 수 있다.
- <28> 2. 최적의 기억 공간 관리
- <29> 트랜스코딩 서버는 기억 공간을 감소된 CPU 사용과 교환할 수 있다. 그러나 기억 공간은 제한된 자원이기 때문에, 현명한 기억 공간 관리 방법이 요구된다. 본 발명에서는, 이와 같은 트레이드-오프를 표현하는 해석적 모델을 제안할 것이다.
- <30> FS_i 를 비디오 V_i 의 저장된 버전들의 모든 실행 가능한 집합들을 원소로 하는 집합이라고 하자. 모든 비디오들의 원본 버전들은 반드시 디스크 상에 저장되어야 한다는 점에 주목하자. 그러므로 만약 버전들의 개수가 NR 이면, FS_i 에 대하여 2^{NR-1} 개의 가능한 집합들이 존재한다. NE 를 집합 FS_i 에서의 원소의 개수라고 하자. 즉, $NE = 2^{NR-1}$.
- <31> FS_i 의 원소들, $FS_{i,j}$ ($j = 1, \dots, NE$)이 기억 공간 요구조건(storage requirement)에 대한 오름차순으로 정렬되어 있다고 가정할 것이다. 예를 들어, 만약 $NR=3$ 이면, $FS_i = \{FS_{i,1} = \{V_i^1\}, FS_{i,2} = \{V_i^1, V_i^2\}, FS_{i,3} = \{V_i^1, V_i^2\}, FS_{i,4} = \{V_i^1, V_i^2, V_i^3\}\}$. 버전 V_i^k 가 디스크 상에 저장되어 있는지 여부를 나타내는 파라미터 $S_{i,j}^k$ 을 다음 수학적 식 2와 같이 정의한다.

수학적 식 2

$$S_{i,j}^k = \begin{cases} 1 & \text{if } V_i^k \in FS_{i,j} \\ 0 & \text{Otherwise.} \end{cases}$$

- <32>
- <33> 비디오 스트림을 트랜스코딩 하기 위해서, 서버는 일반적으로 태스크(task) 또는 쓰레드(thread)를 생성하는데, 이들은 CPU 사이클을 주기적으로 소모한다. 각각의 태스크와 관련된 CPU 이용도(utilization)는 각각의 주기에서 해당 태스크에 할당된 CPU 시간의 비율이다. 버전들이 집합 $FS_{i,j}$ 내의 버전들로서 저장된 경우에, 트랜스코딩 된 버전 V_i^k 을 만들기 위해 요구되는 CPU 이용도를 $C_{i,j}^k$ 라고 하자. 만약 $S_{i,j}^k = 1$ 이면, $C_{i,j}^k = 0$ 이다. 그렇지 않으면, 서버는 $FS_{i,j}$ 내에서 비트-레이트가 다음 단계로 높은 버전(the higher bit-rate version)을 선택하는데, 이것은 트랜스코딩을 위한 최소한의 CPU 이용도를 요구할 것이다.
- <34> TS 를 사용가능한 기억 공간의 전체 양이라고 하자. $STR_{i,j}$ 가 집합 $FS_{i,j}$ 내의 버전들에 대한 전체 기억 공간 요구조건을 나타낸다고 하면, $\sum_{k=1}^{NR} (S_{i,j}^k \times b_i^k \times T_i)$ 와 같이 표현될 수 있다. $CU_{i,j}$ 가 집합 $FS_{i,j}$ 내의 버전들을 저장하기 위해 요구되는 평균 CPU 이용도를 나타낸다고 하면, $\sum_{k=1}^{NR} (p_i^k \times C_{i,j}^k)$ 와 같이 표현될 수 있다. $SV_{i,j}$ 가, 원본 집합 $FS_{i,1}$ 내의 버전들을 저장하는 대신에 집합 $FS_{i,j}$ 내의 버전들을 저장하는 것에 의한 CPU 이용도의 절약치(saving)를 나타낸다고 하면, $SV_{i,j} = CU_{i,1} - CU_{i,j}$. 따라서 $SV_{i,1} = 0$.
- <35> 본원발명의 목적은 기억 공간 제한조건 하에서 각각의 비디오 요구에 대한 CPU 이용도에서의 평균 절약치를 최대화하는 것이다. SE_i 를, FS_i 의 SE_i -번째 원소, FS_{i,SE_i} 가 비디오 V_i 의 버전들의 집합으로서 선택되는 것을 나타내는 선택 파라미터라고 하자. 이 문제를 형식적으로 정의하지는 않을 것이다.
- <36> 정의 1: 버전 선택 문제(Version Selection Problem; VSP)
- <37> 모든 집합 FS_i ($i = 1, \dots, NV$)에 대하여, $\sum_{i=1}^{NV} STR_{i,SE_i} \leq TS$ 을 만족하면서 $\sum_{i=1}^{NV} SV_{i,SE_i}$ 을 최대화하는 SE_i ($SE_i = 1, \dots, NE_i$)

NV)를 찾는다.

<38> VSP는, 각각의 객체가 아이템들의 유한 집합으로 이루어지며, 전체 이익을 최대화하기 위하여 각각의 객체로부터 정확히 하나의 아이템이 선택되어야 하는 다중-선택 배낭 문제의 변형이다. 본 발명에서는 버전 선택 알고리즘(Version Selection Algorithm; VSA)이라고 지칭되는 VSP에 대한 최적해를 제안하는데, 상기 해는 도 2에 도시된 알고리즘 1에서 확인할 수 있는 바와 같이 동적 프로그래밍 기법을 이용한다.

<39> $OPT_{v,w}$ ($1 \leq v \leq NV$, $1 \leq w \leq TS$)를, $V_1 \dots V_v$ 에 대하여 할당된 디스크 기억 공간이 w 인 경우의 최대 CPU 절약치라고 하자. 편의를 위하여, $v=0$ 이고 $w=0$ 인 경우에, $OPT_{0,0}=0$ 으로 설정하자. 만약 $v>0$ 이면, $OPT_{v,w}$ 는 앞서 언급한 바 있는 다음 수학적 식 1에 의하여 계산될 수 있다.

<40> <수학적 식 1>

$$OPT_{v,w} = \max_{1 \leq j \leq NE} \{ \max \{ OPT_{v-1,w}, OPT_{v-1,w-STR_{v,j}} + SV_{v,j} \} \}$$

<42> $OPT_{NV,TS}$ 의 값은 사용가능한 디스크 기억 공간이 TS 인 경우의 CPU 이용도에서의 최대 절약치이다. 다음으로, SE_i 가 다음과 같이 계산될 수 있다. 즉, 각각의 반복에서, VSA는 CPU 이용도에서의 가장 큰 절약치에 해당하는 원소의 인덱스를 저장한다(알고리즘 1의 제13행). 다음으로, $OPT_{NV,TS}$ 로부터 더듬어 올라가는 것에 의해 SE_i 를 검색한다(알고리즘 1의 제18-23행). 알고리즘 1이 $O(NV \times TS \times NE)$ 시간으로 동작한다는 것을 쉽게 확인할 수 있다.

<43> 3. 실험 결과

<44> 본 발명에서 제안하는 방법의 효율성을 평가하기 위해, 몇 가지 시뮬레이션을 수행하였다. 5개의 샘플 비디오(예컨대, 각각 한 편씩의 다큐멘터리, 애니메이션, 드라마, 축구 시합, 교육용 영화)를 트랜스코딩 하기 위해 요구되는 CPU 이용도를, 인텔 펜티엄 3GHz 프로세서에 기초한 Linux 머신 상에서 FFmpeg 프로그램을 실행시킴으로써 측정하였다. 표 1은 5개의 샘플 비디오 중 하나의 트랜스코딩에 요구되는 CPU 이용도 및 기억 공간 요구조건을 나타낸다. 각각의 버전에 대하여 기억 공간의 최소 유닛이 메가바이트인 것으로 가정한다.

표 1

기억 공간 요구조건		CPU 이용도			
해상도	크기(MB)	트랜스코딩	CPU 이용도	트랜스코딩	CPU 이용도
VGA (640× 480)	115	VGA → CIF	8 %	CIF → QCIF	3.9 %
CIF (352× 288)	94	VGA → QVGA	7.5 %	CIF → SQCIF	3.3 %
QVGA (320× 240)	85	VGA → QCIF	6.4 %	QVGA → QCIF	3.8 %
QCIF (176× 144)	76	VGA → SQCIF	5.8 %	QVGA → SQCIF	3.2 %
SQCIF (128× 96)	60	CIF → QVGA	4.8 %	QCIF → SQCIF	3.1 %

<46> 각각의 샘플 비디오의 원본 버전을 200번 저장하고, 이들을 서로 다른 비디오로 간주함으로써 1000개의 비디오를 제공하는 시스템을 시뮬레이션하였다. 이들 1000개의 비디오에 대한 최소 기억 공간은 428 GB이다. NR은 5로 선택된다. 고객 요구의 도착은, 3초의 평균 내부-도착 시간을 갖는 Poisson 분포를 따르는 것으로 간주된다. 접근 확률은, $\alpha = 0.271$ 을 갖는 Zipf 분포를 따른다. VSA 스킴을 평가하기 위하여, 다음과 같은 4개의 다른 방법들과 비교할 것이다.

<47> 1) OBS: 이 방법은 비디오들의 원본 버전만을 저장한다.

<48> 2) PBS: 이 방법은 각각의 비디오의 인기도(popularity)에 기초하여 디스크 상에 저장될 버전들을 선택한다. 기억 공간이 다 소모될 때까지, 가장 인기 있는 비디오부터 다음으로 인기 있는 비디오 등의 모든 트랜스코딩된 버전을 저장한다.

<49> 3) VBS: 이 알고리즘은 보다 높은 비트-레이트의 버전을 선호한다. 가장 인기 있는 비디오, 다음으로 인기 있는 비디오 등의 가장 높은 비트-레이트로 트랜스코딩 된 버전을 저장한다. 가장 높은 비트-레이트로 트랜스코딩 된 모든 버전들이 저장되면, 다음으로 높은 비트-레이트의 버전에 대하여 동일한 처리를 반복한다. 이와 같은 과정을 기억 공간이 소진될 때까지 반복한다.

- <50> 4) RBS: 이 방법은 임의의 비디오의 임의의 버전을 랜덤으로 선택하여 디스크 상에 저장한다. 이러한 과정을 기억 공간이 소진될 때까지 반복된다.
- <51> 도 3은, 비디오의 모든 버전의 접근 확률이 동일한 경우에, 즉 $\forall i, p_i^k = p_i \times \frac{1}{NR}, i = 1, \dots, NV$ 인 경우에 서비스를 제공받는 고객들의 비율인 인가 비율(admission ratio)이 기억 공간의 전체 양(TS)에 얼마나 영향을 받는지를 도시한다. 모든 작업부하 조건 하에서, VSA 스킵이 최고의 성능을 나타낸다. 즉, VSA 스킵은 OBS보다 26% 내지 50% 사이의 고객을 더 인가하고, PBS보다 5% 내지 6% 사이의 고객을 더 인가하며, VBS보다 14% 내지 20% 사이의 고객을 더 인가하고, RBS보다 12% 내지 14% 사이의 고객을 더 인가한다. TS가 증가함에 따라 서버는 보다 많은 트랜스코딩 된 버전들을 저장할 수 있으며, 이것은 트랜스코딩에 대한 CPU 상의 수요를 줄이기 때문에, 인가 비율이 증가한다.
- <52> 도 4는, TS = 800 GB인 경우에, 서로 다른 버전들에 대한 요구의 분포가 인가 비율에 얼마나 영향을 받는지를 도시한다. 이와 같은 목적을 위하여, 다음과 같은 3가지 상황을 정의한다.
- <53> HRM: 가장 높은 해상도의 버전이 가장 인기가 높다.
- <54> $\forall i, p_i^1 = p_i \times 0.6, p_i^2 = p_i^3 = p_i^4 = p_i^5 = p_i \times 0.1$
- <55> MRM: 중간 해상도의 버전이 가장 인기가 높다.
- <56> $\forall i, p_i^3 = p_i \times 0.6, p_i^1 = p_i^2 = p_i^4 = p_i^5 = p_i \times 0.1$
- <57> LRM: 가장 낮은 해상도의 버전이 가장 인기가 높다.
- <58> $\forall i, p_i^5 = p_i \times 0.6, p_i^1 = p_i^2 = p_i^3 = p_i^4 = p_i \times 0.1$
- <59> VSA 스킵은 모든 작업 부하 조건 하에서 최고의 성능을 나타낸다. 즉, VSA 스킵은 OBS보다 22% 내지 68% 사이의 고객을 더 인가하고, PBS보다 3% 내지 26% 사이의 고객을 더 인가하며, VBS보다 10% 내지 51% 사이의 고객을 더 인가하고, RBS보다 6% 내지 34% 사이의 고객을 더 인가한다. VSA 스킵에서, 저해상도 버전들에 대한 요구가 더 많을 경우 인가 비율이 감소한다. 그러나 MRM 상황에서 다른 스킵들은 더 나쁜 성능을 나타내는데, 이것은 다음과 같이 설명될 수 있다. 즉, 표 1로부터, 저해상도 버전들보다 중간-해상도 버전들을 추출하는데 보다 많은 CPU 타임이 소비된다는 것을 확인할 수 있는데, VSA 스킵은 디스크 상에 저장될 버전들을 선택하는 데 있어 이러한 차이를 고려하지만, 다른 스킵들은 그렇지 못하다는 것이다.
- <60> 본 발명은, 트랜스코딩이 가능한 멀티미디어 서버에 있어서 QoS-인지 비디오 스트리밍에 대하여 최적의 기억 공간 관리 스킵을 제안하였다. 본 발명은, 트랜스코딩에 대한 CPU 수요를 최소화하기 위한 목적으로, 디스크 상에 저장될 버전들을 선택하기 위하여 일반화된 해석 모델을 사용한다. 이후, 본 발명은, 디스크 상에 저장될 버전들을 최적으로 선택하는 알고리즘을 제안한다. 실험 결과로부터, 본 발명에서 제안한 방법이 트랜스코딩에 의한 CPU 상의 수요들을 실질적으로 감소시키며, 그 결과 서버가 이중 QoS 요구조건을 만족시키면서 보다 많은 고객들을 인가할 수 있도록 한다는 것을 확인할 수 있다.
- <61> 이상 설명한 본 발명은 본 발명이 속한 기술분야에서 통상의 지식을 가진 자에 의하여 다양한 변형이나 응용이 가능하며, 본 발명에 따른 기술적 사상의 범위는 아래의 특허청구범위에 의하여 정해져야 할 것이다.

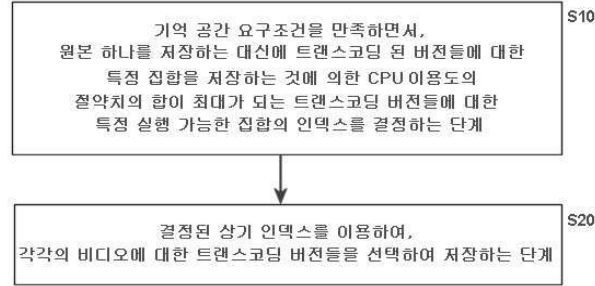
도면의 간단한 설명

- <62> 도 1은 본 발명의 일 실시예에 따른, 트랜스코딩 서버의 기억 공간 관리 방법의 구성을 나타내는 도면.
- <63> 도 2는 본 발명의 일 실시예에 따른 트랜스코딩 서버의 기억 공간 관리 방법을 구현한 알고리즘의 일 예를 나타내는 도면.
- <64> 도 3은, 비디오의 모든 버전의 접근 확률이 동일한 경우에, 서비스를 제공받는 고객들의 비율인 인가 비율(admission ratio)이 기억 공간의 전체 양(TS)에 얼마나 영향을 받는지를 나타내는 도면.
- <65> 도 4는, TS = 800 GB인 경우에, 서로 다른 버전들에 대한 요구의 분포가 인가 비율에 얼마나 영향을 받는지를 나타내는 도면.
- <66> <도면 중 주요 부분에 대한 부호의 설명>

- <67> S10: 기억 공간 요구조건을 만족하면서 원본 하나를 저장하는 대신에 트랜스코딩 버전들에 대한 특정 실행 가능한 집합을 저장하는 것에 의한 CPU 이용도의 절약치의 합이 최대가 되는 트랜스코딩 버전들에 대한 특정 실행 가능한 집합의 인덱스를 결정하는 단계
- <68> S20: 단계 S10에서 결정된 인덱스를 이용하여 각각의 비디오에 대한 트랜스코딩 버전들을 선택하여 저장하는 단계

도면

도면1



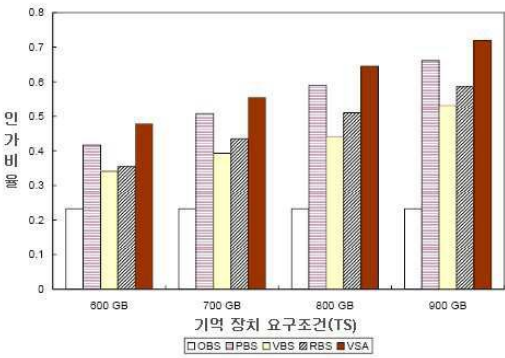
도면2

Algorithm 1 VSA (version selection algorithm)

```

1: Temporary variables:  $w, v, j$ ;
2: for  $w = 0$  to  $TS$  do
3:   for  $v = 0$  to  $NV$  do
4:      $OPT_{v,w} \leftarrow 0$ ;
5:      $OPT\_SET_{v,w} \leftarrow 1$ ;
6:   end for
7: end for
8: for  $v = 1$  to  $NV$  do
9:   for  $w = TS$  downto  $STR_{v,1}$  do
10:    for  $j = 1$  to  $NE$  do
11:      if  $OPT_{v-1,w-STR_{v,j}} + SV_{v,j} > OPT_{v-1,w}$  then
12:         $OPT_{v,w} \leftarrow OPT_{v-1,w-STR_{v,j}} + SV_{v,j}$ ;
13:         $OPT\_SET_{v,w} \leftarrow j$ ;
14:      end if
15:    end for
16:  end for
17: end for
18:  $w \leftarrow TS, v \leftarrow NV$ ;
19: while  $v > 0$  do
20:    $SE_v \leftarrow OPT\_SET_{v,w}$ ;
21:    $w \leftarrow w - STR_{v,SE_v}$ ;
22:    $v \leftarrow v - 1$ ;
23: end while
  
```

도면3



도면4

