(19) 대한민국특허청(KR)
(12) 공개특허공보(A)

(11) 공개번호 10-2008-0025399
(43) 공개일자 2008년03월20일

(51) Int. Cl.
G10L 19/00 (2006.01) *G10L 19/02* (2006.01)
G10L 15/00 (2006.01)
(21) 출원번호 10-2008-7000635
(22) 출원일자 2008년01월09일
심사청구일자 없음
번역문제출일자 2008년01월09일
(86) 국제출원번호 PCT/US2006/027231
국제출원일자 2006년07월14일
(87) 국제공개번호 WO 2007/011653
국제공개일자 2007년01월25일
(30) 우선권주장
11/183,266 2005년07월15일 미국(US)

(71) 출원인
마이크로소프트 코포레이션
미국 워싱턴주 (우편번호 : 98052) 레드몬드 원 마이크로소프트 웨이
(72) 발명자
매로트라, 산지브
미국 98052-6399 워싱톤주 레드몬드 원 마이크로소프트 웨이
첸, 웨이-게
미국 98052-6399 워싱톤주 레드몬드 원 마이크로소프트 웨이
(74) 대리인
양영준, 백만기

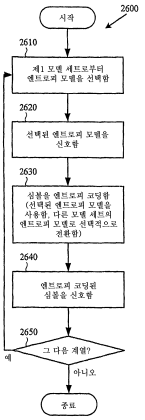
전체 청구항 수 : 총 20 항

(54) 적응적 코딩 및 디코딩에서 다수의 엔트로피 모델의 선택적사용

(57) 요 약

적응적 코딩 및 디코딩에서 다수의 엔트로피 모델을 선택적으로 사용하는 기법 및 도구가 본 명세서에 기술되어 있다. 예를 들어, 다수의 심볼에 대해, 오디오 인코더는 다수의 엔트로피 모델을 포함하는 제1 모델 세트로부터 엔트로피 모델을 선택한다. 다수의 엔트로피 모델 각각은 하나 이상의 엔트로피 모델을 포함하는 제2 모델 세트로 전환하기 위한 모델 전환점(model switch point)을 포함한다. 인코더는 선택된 엔트로피 모델을 사용하여 다수의 심볼을 처리하고 결과를 출력한다. 엔트로피 모델을 발생하는 기법 및 도구도 기술되어 있다.

대표도 - 도26



특허청구의 범위

청구항 1

복수의 심볼에 대해, 다수의 엔트로피 모델을 포함하는 제1 모델 세트로부터 엔트로피 모델을 선택하는 단계 - 상기 제1 모델 세트의 다수의 엔트로피 모델 각각은 하나 이상의 엔트로피 모델을 포함하는 제2 모델 세트로의 전환을 위한 모델 전환점을 포함함 -,

상기 선택된 엔트로피 모델을 사용하여 상기 복수의 심볼을 처리하는 단계, 및

상기 처리의 결과를 출력하는 단계를 포함하는 방법.

청구항 2

제1항에 있어서, 인코더가 인코딩 동안에 상기 선택하는 단계, 상기 처리하는 단계, 및 상기 출력하는 단계를 수행하며,

상기 처리는 엔트로피 인코딩을 포함하는 것인 방법.

청구항 3

제1항에 있어서, 디코더가 디코딩 동안에 상기 선택하는 단계, 상기 처리하

는 단계, 및 상기 출력하는 단계를 수행하며,

상기 처리는 엔트로피 디코딩을 포함하는 것인 방법.

청구항 4

제1항에 있어서, 상기 제1 모델 세트의 다수의 엔트로피 모델 및 상기 제2 모델 세트의 하나 이상의 엔트로피 모델은 산술 코딩 및/또는 디코딩에 대한 확률 분포이고,

상기 모델 전환점은 상기 제1 모델 세트의 다수의 확률 분포에서의 모델 전환 확률인 것인 방법.

청구항 5

제1항에 있어서, 상기 제1 모델 세트의 다수의 엔트로피 모델은 각각 제1 테이블 세트의 다수의 VLC 테이블로 구현되고,

상기 제2 모델 세트의 하나 이상의 엔트로피 모델은 각각 제2 테이블 세트의 하나 이상의 VLC 테이블로 구현되며,

상기 모델 전환점은 이스케이프 코드이고,

상기 제1 테이블 세트의 다수의 VLC 테이블 각각은 상기 제2 테이블 세트로 전환하기 위한 상기 이스케이프 코드를 포함하는 것인 방법.

청구항 6

제5항에 있어서, 상기 제1 테이블 세트의 다수의 VLC 테이블 및 상기 제2 테이블 세트의 하나 이상의 VLC 테이블은 허프만 코드 테이블이고,

상기 제2 테이블 세트는 하나의 허프만 코드 테이블을 포함하며, 그에 따라 이 하나의 허프만 코드 테이블이 상기 제1 테이블 세트의 각자의 다수의 허프만 코드 테이블을 나타내는 트리에서의 공통 가지를 나타내는 것인 방법.

청구항 7

제5항에 있어서, 상기 제1 테이블 세트의 다수의 VLC 테이블은 확률이 높은 심볼값을 포함하는 제1 심볼값 세트에 대해 적응되어 있고,

상기 제2 테이블 세트의 하나 이상의 VLC 테이블은 확률이 낮은 심볼값을 포함하는 제2 심볼값 세트에 대해 적

응되어 있는 것인 방법.

청구항 8

제7항에 있어서, 상기 제2 테이블 세트는 하나의 VLC 테이블을 포함하고,

상기 처리는 상기 확률이 낮은 심볼값을 갖는 복수의 심볼의 값의 2-단계 가변 길이 코딩 또는 디코딩을 위한 것인 방법.

청구항 9

제1항에 있어서, 상기 제1 모델 세트의 다수의 엔트로피 모델 및 상기 제2 모델 세트의 하나 이상의 엔트로피 모델을 발생하는 단계를 더 포함하며,

상기 발생하는 단계는,

제1 비용 메트릭에 따라 확률 분포를 클러스터링하고, 그 결과 복수의 예비 클러스터를 얻는 단계, 및

상기 제1 비용 메트릭과 다른 제2 비용 메트릭에 따라 상기 복수의 예비 클러스터를 세분하고, 그 결과 복수의 최종 클러스터를 얻는 단계를 포함하는 것인 방법.

청구항 10

제1항에 있어서, 상기 제2 모델 세트는 하나의 엔트로피 모델을 포함하고,

상기 방법은,

상기 제1 모델 세트의 다수의 엔트로피 모델 및 상기 제2 모델 세트의 하나의 엔트로피 모델을 발생하는 단계를 더 포함하며,

상기 발생하는 단계는, 상기 제2 모델 세트의 상기 하나의 엔트로피 모델에 대해, 확률 분포들에 걸쳐 공통의 조건부 분포를 갖도록 확률이 낮은 심볼값을 제약하는 단계를 포함하는 것인 방법.

청구항 11

제1항에 있어서, 상기 제2 모델 세트의 하나 이상의 엔트로피 모델 각각은 하나 이상의 엔트로피 모델을 포함하는 제3 모델 세트로의 전환을 위한 제2 모델 전환점을 포함하는 것인 방법.

청구항 12

제1항에 있어서, 상기 제1 모델 세트의 다수의 엔트로피 모델 중 적어도 일부에 대해, 상기 모델 전환점이 모델마다 서로 다른 값을 갖는 것인 방법.

청구항 13

제1항에 있어서, 상기 제1 모델 세트의 다수의 엔트로피 모델 각각은 하나 이상의 엔트로피 모델을 포함하는 제3 모델 세트로의 전환을 위한 제2 모델 전환점을 더 포함하는 것인 방법.

청구항 14

제1항에 있어서, 상기 복수의 심볼은 오디오 데이터의 양자화된 스펙트럼 계수에 대한 것인 방법.

청구항 15

제1항에 있어서, 상기 선택하는 단계는 전방 적응적 전환의 일부인 것인 방법.

청구항 16

제1항에 있어서, 상기 선택하는 단계는 후방 적응적 전환의 일부인 것인 방법.

청구항 17

제1 비용 메트릭에 따라 확률 분포를 클러스터링하고, 그 결과 복수의 예비 클러스터를 얻는 단계,

상기 제1 비용 메트릭과 다른 제2 비용 메트릭에 따라 상기 복수의 예비 클러스터를 세분하고, 그 결과 복수의 최종 클러스터를 얻는 단계, 및

상기 최종 클러스터에 적어도 부분적으로 기초하여 상기 엔트로피 모델을 설정하는 단계에 의해,

엔트로피 모델을 발생하는 하나 이상의 모델을 포함하는 시스템.

청구항 18

제17항에 있어서, 상기 제2 비용 메트릭은 상대 엔트로피인 것인, 엔트로피 모델을 발생하는 하나 이상의 모델을 포함하는 시스템.

청구항 19

심볼값에 대한 확률 분포를 구하는 수단, 및

상기 확률 분포에 걸쳐 공통의 조건부 분포를 갖도록 복수의 확률이 낮은 심볼값을 제약하고 복수의 확률이 높은 심볼값을 그렇게 제약하지 않는 것을 포함하는, 엔트로피 모델을 발생하는 수단을 포함하는 시스템.

청구항 20

제19항에 있어서, 상기 엔트로피 모델은 각각 제1 테이블 세트의 다수의 VLC 테이블 및 제2 테이블 세트의 하나의 VLC 테이블로 구현되고,

상기 다수의 VLC 테이블은 상기 복수의 확률이 높은 심볼값에 대해 적응되고,

상기 하나의 VLC 테이블은 상기 복수의 확률이 낮은 심볼값에 대해 적응되는 것인, 엔트로피 모델을 발생하는 수단을 포함하는 시스템.

명 세 서

배 경 기 술

<1> 엔지니어들은 디지털 오디오의 품질을 여전히 유지하면서 디지털 오디오를 효율적으로 처리하기 위해 다양한 기법들을 사용한다. 이들 기법을 이해하기 위해서는, 오디오 정보가 컴퓨터에서 어떻게 표현되고 처리되는지를 이해하는 것이 도움이 된다.

<2> **I. 컴퓨터에서의 오디오 정보의 표현**

<3> 컴퓨터는 오디오 정보를 오디오 정보를 표현하는 일련의 숫자로서 처리한다. 예를 들어, 단일의 숫자는 특정의 시간에서의 진폭값인 오디오 샘플을 표현할 수 있다. 샘플 깊이(sample depth), 샘플링 레이트(sampling rate) 및 채널 모드(channel mode)를 비롯한 몇가지 인자가 오디오 정보의 품질에 영향을 준다.

<4> 샘플 깊이(또는 정밀도(precision))는 샘플을 표현하는 데 사용되는 숫자들의 범위를 나타낸다. 샘플에 대해 가능한 값들이 많을수록, 품질이 더 높아지는데, 그 이유는 그 숫자가 더 미묘한 진폭의 변동을 포착할 수 있기 때문이다. 예를 들어, 8-비트 샘플은 256개의 가능한 값을 갖는 반면, 16-비트 샘플은 65,536개의 가능한 값을 갖는다.

<5> 샘플링 레이트(보통 초당 샘플의 수로 측정됨)도 품질에 영향을 미친다. 샘플링 레이트가 높을수록, 품질이 더 높은데, 그 이유는 더 많은 소리 주파수가 표현될 수 있기 때문이다. 어떤 통상적인 샘플링 레이트는 8,000, 11,025, 22,050, 32,000, 44,100, 48,000 및 96,000 샘플/초이다.

<6> 모노 및 스테레오는 2가지 통상적인 오디오 채널 모드이다. 모노 모드에서, 오디오 정보는 하나의 채널에 존재한다. 스테레오 모드에서, 오디오 정보는 통상 좌채널 및 우채널이라고 하는 2개의 채널에 존재한다. 5.1 채널, 7.1 채널, 또는 9.1 채널 서라운드 사운드("1"은 서브-우퍼 또는 저주파 효과 채널을 가리킴) 등의 더 많은 채널을 갖는 다른 모드들도 가능하다. 표 1은 서로 다른 품질 레벨을 갖는 몇가지 오디오 형식을, 대응하는 원시 비트 레이트 비용(raw bit rate cost)과 함께 나타낸 것이다.

<7> <표 1> 서로 다른 품질의 오디오 정보에 대한 비트 레이트

<8>

	샘플 깊이 (비트/샘플)	샘플링 레이트 (샘플/초)	채널 모드	원시 비트 레이트 (비트/초)
인터넷 전화	8	8,000	모노	64,000
전화	8	11,025	모노	88,200
CD 오디오	16	44,100	스테레오	1,411,200

<9> 서라운드 사운드 오디오는 일반적으로 훨씬 더 높은 원시 비트 레이트를 갖는다. 표 1이 보여주는 바와 같이, 고품질 오디오 정보의 대가는 고 비트 레이트이다. 고품질 오디오 정보는 많은 양의 컴퓨터 저장장치 및 전송 용량을 차지한다. 그렇지만, 회사 및 소비자는 고품질 오디오 컨텐츠를 생성, 배포 및 재생하는 데 점점 더 컴퓨터에 의존한다.

<10> **II. 컴퓨터에서의 오디오 정보의 처리**

<11> 많은 컴퓨터 및 컴퓨터 네트워크는 원시 디지털 오디오를 처리하기에는 자원이 부족하다. 압축(compression)(인코딩 또는 코딩이라고도 함)은 오디오 정보를 더 낮은 비트 레이트 형태로 변환함으로써 그 정보를 저장 및 전송하는 비용을 감소시킨다. 압축은 무손실(lossless)(품질이 나빠지지 않음)이거나 손실(lossy)(품질이 나빠지지만 차후의 무손실 압축으로부터의 비트 레이트 감소가 더 극적임)일 수 있다. 예를 들어, 손실 압축은 원래의 오디오 정보의 근사값을 구하는 데 사용되며, 이 근사값이 이어서 무손실 압축된다. 압축 해제(decompression)(디코딩이라고도 함)는 압축된 형태로부터 원래의 정보의 재구성된 버전을 추출한다.

<12> 오디오 압축의 한 목적은 가능한 최소량의 비트로 최대의 지각 신호 품질을 제공하기 위해 오디오 신호를 디지털적으로 표현하는 것이다. 이 목적을 목표로 삼아, 다양한 최근의 오디오 인코딩 시스템은 사람의 지각 모델을 사용한다. 인코더 및 디코더 시스템은 마이크로소프트사의 WMA(Windows Media Audio) 인코더 및 디코더와 WMA Pro 인코더 및 디코더의 어떤 버전들을 포함한다. 다른 시스템들은 MP3(Motion Picture Experts Group, Audio Layer 3) 표준, MPEG2(Motion Picture Experts Group 2), AAC(Advanced Audio Coding) 표준, 및 돌비 AC3의 어떤 버전들에 의해 규정된다. 이러한 시스템들은 일반적으로 손실/무손실 압축/압축해제의 조합을 사용한다.

<13> **A. 손실 압축 및 대응하는 압축해제**

<14> 종래에, 오디오 인코더는 다양한 서로 다른 손실 압축 기법을 사용한다. 이들 손실 압축 기법은 일반적으로 주파수 변환 이후에 지각 모델링/가중 및 양자화를 수반한다. 대응하는 압축해제는 역양자화, 역가중화, 및 역주파수 변환을 수반한다.

<15> 주파수 변환 기법은 데이터를 지각적으로 중요한 정보를 지각적으로 중요하지 않은 정보와 분리시키는 것을 더 용이하게 만들어주는 형태로 변환한다. 이어서, 주어진 비트 레이트에 대해 최상의 지각 품질을 제공하기 위해, 덜 중요한 정보가 더 많은 손실 압축을 겪을 수 있는 반면, 더 중요한 정보는 보존된다. 주파수 변환은 일반적으로 오디오 샘플을 수신하고 이들을 주파수 영역의 데이터(때때로, 주파수 계수 또는 스펙트럼 계수라고 함)로 변환한다.

<16> 지각 모델링은 주어진 비트 레이트에 대해 재구성된 오디오 신호의 지각 품질을 향상시키기 위해 사람의 청각 시스템의 모델에 따라 오디오 데이터를 처리하는 것을 수반한다. 지각 모델링의 결과를 사용하여, 인코더는 주어진 비트 레이트에 대해 노이즈의 가청도(audibility)를 최소화하기 위해 오디오 데이터에서의 노이즈(예를 들어, 양자화 노이즈)를 쉐이핑(shape)한다.

<17> 양자화는 입력값의 범위를 단일의 값으로 매핑하고 정보의 비가역 손실을 가져오지만 인코더가 출력의 품질 및 비트 레이트를 조절할 수 있게도 해준다. 때때로, 인코더는 비트 레이트 및/또는 품질을 조절하기 위해 양자화를 조정하는 레이트 제어기와 함께 양자화를 수행한다. 적응적(adaptive) 및 비적응적(non-adaptive), 스칼라(scalar) 및 벡터(vector), 균일(uniform) 및 비균일(non-uniform)을 비롯한 다양한 종류의 양자화가 있다. 지각 가중(perceptual weighting)은 비균일 양자화의 한 형태로 간주될 수 있다.

<18> 역양자화 및 역가중은 가중되고 양자화된 주파수 계수 데이터를 원래의 주파수 계수 데이터의 근사값으로 재구성한다. 이어서, 역주파수 변환은 재구성된 주파수 계수 데이터를 재구성된 시간 영역 오디오 샘플로 변환한다.

<19> **B. 무손실 압축 및 압축해제**

<20> 종래에, 오디오 인코더는 다양한 서로 다른 무손실 압축 기법(엔트로피 코딩 기법이라고도 함) 중 하나 이상을 사용한다. 일반적으로, 무손실 압축 기법은 런-길이 인코딩(run-length encoding), 가변 길이 인코딩(variable length encoding), 및 산술 코딩(arithmetic coding)을 포함한다. 대응하는 압축해제 기법(엔트로피 디코딩 기법이라고도 함)은 런-길이 디코딩, 가변 길이 디코딩, 및 산술 디코딩을 포함한다.

<21> 런-길이 인코딩은 간단한 잘 알려진 압축 기법이다. 일반적으로, 런-길이 인코딩은 동일한 값을 갖는 연속적인 심볼의 시퀀스(즉, 런(run))를 그 시퀀스의 값 및 길이로 대체시키는 것이다. 런-길이 디코딩에서, 연속적인 심볼의 시퀀스가 런 값(run value) 및 런 길이(run length)로부터 재구성된다. 런-길이 인코딩 및 디코딩의 수많은 변형들이 개발되었다.

<22> 런-레벨 인코딩(run-level encoding)은 동일한 값을 갖는 연속적인 심볼의 런이 런 길이로 대체된다는 점에서 런-길이 인코딩과 유사하다. 런의 값은 데이터에서의 우세값(predominant value)(예를 들어, 0)이고, 런은 다른 값(예를 들어, 영이 아닌 값)을 갖는 하나 이상의 레벨에 의해 분리된다.

<23> 런-길이 인코딩의 결과(예를 들어, 런 값 및 런 길이) 또는 런-레벨 인코딩의 결과는 비트 레이트를 추가적으로 감소시키기 위해 가변 길이 코딩(variable length coding)될 수 있다. 그렇게 한 경우, 가변 길이 코딩된 데이터는 런-길이 디코딩 이전에 가변 길이 디코딩된다.

<24> 가변 길이 코딩은 다른 잘 알려진 압축 기법이다. 일반적으로, VLC(variable length code, 가변 길이 코드) 테이블은 VLC를 고유의 심볼값(값들의 고유의 조합)과 연관시킨다. 허프만 코드는 통상적인 유형의 VLC이다. 확률이 더 높은 심볼값에 더 짧은 코드가 할당되고, 확률이 더 낮은 심볼값에 더 긴 코드가 할당된다. 어떤 종류의 컨텐츠의 대표적인 예에 대해 이 확률이 계산된다. 또는, 막 인코딩된 데이터에 대해 또는 인코딩될 데이터에 대해 이 확률이 계산되며, 이 경우 VLC는 고유의 심볼값에 대한 변하는 확률에 적응한다. 정적 가변 길이 코딩(static variable length coding)과 비교하여, 적응적 가변 길이 코딩(adaptive variable length coding)은 보통 데이터에 대한 더 정확한 확률을 구현함으로써 압축된 데이터의 비트 레이트를 감소시키지만, VLC를 규정하는 추가 정보도 전송될 필요가 있을 수 있다.

<25> 심볼을 인코딩하기 위해, 가변 길이 인코더는 심볼값을 VLC 테이블에 있는 심볼값과 연관된 VLC로 대체시킨다. 디코딩하기 위해, 가변 길이 디코더는 VLC를 VLC와 연관된 심볼값으로 대체시킨다.

<26> 스칼라 가변 길이 코딩에서, VLC 테이블은 단일의 VLC를 하나의 값, 예를 들어, 양자화된 데이터 값의 직류 레벨(direct level)과 연관시킨다. 벡터 가변 길이 코딩에서, VLC 테이블은 단일의 VLC를, 값들의 조합, 예를 들어, 양자화된 데이터 값의 직류 레벨의 그룹과 특정의 순서로 연관시킨다. 벡터 가변 길이 인코딩은 (예를 들어, 인코더가 이진 VLC에서 확률을 아주 세분하여 이용할 수 있게 해줌으로써) 스칼라 가변 길이 인코딩보다 더 나은 비트 레이트 감소를 가져올 수 있다. 반면에, 단일의 코드가 많은 그룹의 심볼을 표현하거나 심볼이 (많은 수의 가능한 조합으로 인해) 큰 범위의 가능한 값을 가질 때, 벡터 가변 길이 인코딩에 대한 VLC 테이블이 극단적으로 클 수 있으며, 이는 VLC 테이블을 계산하고 VLC를 찾는 데 메모리 및 처리 자원을 소비한다. 가변 길이 인코딩/디코딩의 수많은 변형들이 개발되었다.

<27> 산술 코딩은 또하나의 잘 알려진 압축 기법이다. 산술 코딩은 때때로 주어진 입력 심볼을 인코딩하는 최적의 비트수가 분수 비트수(a fractional number of bits)인 응용에서 및 어떤 개개의 입력 심볼들 간의 통계적 상관이 존재하는 경우에 사용된다. 산술 코딩은 일반적으로 입력 시퀀스를 주어진 범위 내의 단일의 수로 표현하는 것을 수반한다. 일반적으로, 이 수는 0과 1 사이의 분수이다. 입력 시퀀스 내의 심볼은 0과 1 사이의 공간의 일부분을 차지하는 범위와 연관된다. 이 범위는 특정의 심볼이 입력 시퀀스에 있을 확률에 기초하여 계산된다. 입력 시퀀스를 표현하는 데 사용되는 분수는 범위를 참조하여 해석된다. 따라서, 산술 코딩 방식에서는 입력 심볼에 대한 확률 분포가 중요하다.

<28> 컨텍스트-기반 산술 코딩(context-based arithmetic coding)에서, 입력 심볼에 대한 서로 다른 확률 분포가 서로 다른 컨텍스트와 연관된다. 컨텍스트가 변할 때, 입력 시퀀스를 인코딩하는 데 사용되는 확률 분포가 변한다. 이 컨텍스트는 특정의 입력 심볼이 입력 시퀀스에 나타날 확률에 영향을 줄 것으로 예상되는 서로 다른 인자를 측정함으로써 계산된다.

<29> 미디어 처리에 대한 압축 및 압축해제의 중요성을 고려하면, 압축 및 압축해제가 풍부하게 개발된 분야라는 것은 놀랍지 않다. 종래 기술의 무손실 압축 및 압축해제 기법 및 시스템의 이점이 무엇이든지간에, 이들은 본

명세서에 기술된 기법 및 시스템의 다양한 이점을 갖지 않는다.

발명의 상세한 설명

<30> 적응적 코딩 및 디코딩에서 다수의 엔트로피 모델을 선택적으로 사용하는 기법 및 도구가 본 명세서에 기술되어 있다. 예를 들어, 다수의 엔트로피 모델을 선택적으로 사용하는 것은 다수의 분포/VLC 테이블을 위한 자원 사용을 상당히 감소시킬 수 있다. 동시에, 다수의 분포/VLC 테이블을 사용하는 것과 연관된 인코딩 이득의 대부분이 달성될 수 있다.

<31> 제1 일련의 기법 및 도구에 따르면, 심볼에 대한 인코더 또는 디코더 등의 도구는 다수의 엔트로피 모델을 포함하는 제1 모델 세트로부터 엔트로피 모델을 선택한다. 제1 모델 세트의 다수의 엔트로피 모델 각각은 하나 이상의 엔트로피 모델을 포함하는 제2 모델 세트로 전환하기 위한 모델 전환점을 포함한다. 이 도구는 선택된 엔트로피 모델을 사용하여 심볼을 처리하고 이 처리의 결과를 출력한다.

<32> 제2 모델 세트의 하나 이상의 엔트로피 모델 각각은 그 자체가 다른 모델 세트로 전환하기 위한 모델 전환점을 포함할 수 있다. 게다가, 제1 모델 세트의 다수의 엔트로피 모델 각각은 다른 모델 세트로 전환하기 위한 제2 모델 전환점을 더 포함할 수 있다. 보다 일반적으로, 제1 모델 세트의 다수의 엔트로피 모델 각각은 다른 모델 세트(들)(다른 모델 세트(들) 중의 각각의 세트는 그 자체가 0개 이상의 엔트로피 모델을 포함하고 있음)로 전환하기 위한 0개 이상의 모델 전환점을 포함할 수 있다. 재귀적 방식으로, 다른 모델 세트(들) 중의 주어진 모델 세트에 대해, 그 모델 세트에 대한 엔트로피 모델(들)은 또다른 모델 세트(들)로 전환하기 위한 0개 이상의 모델 전환점을 포함할 수 있으며, 이하 마찬가지이다.

<33> 제2 일련의 기법 및 도구에 따르면, 시스템은 엔트로피 모델을 발생한다. 이 시스템은 제1 비용 메트릭(cost metric)(평균 제곱 오차(mean squared error) 등)에 따라 확률 분포들을 클러스터링하여, 그 결과 예비 클러스터(preliminary cluster)가 얻어진다. 이 시스템은 제1 비용 메트릭과 다른 제2 비용 메트릭(상대 엔트로피(relative entropy) 등)에 따라 예비 클러스터를 세분화하고, 그 결과 최종 클러스터(final cluster)가 얻어진다. 이 시스템은 이어서 적어도 부분적으로 최종 클러스터에 기초하여 엔트로피 모델을 설정한다.

<34> 제3 일련의 기법 및 도구에 따르면, 시스템은 심볼값에 대한 확률 분포를 구한다. 이 시스템은 엔트로피 모델을 발생한다. 그렇게 함에 있어서, 이 시스템은 다수의 확률이 낮은 심볼값을 확률 분포에 걸쳐 공통의 조건부 분포(common conditional distribution)를 갖도록 제약하고 다수의 확률이 높은 심볼값을 그렇게 제약하지 않는다.

<35> 본 발명의 상기한 목적, 특징 및 이점과 기타의 목적, 특징 및 이점이 첨부 도면을 참조하여 계속되는 이하의 상세한 설명으로부터 보다 명백하게 될 것이다.

실 시 예

<53> 엔트로피 코딩/디코딩 및 연관된 처리를 위한 다양한 기법 및 도구가 기술되어 있다. 이들 기법 및 도구는 아주 낮은 비트 레이트에서도 고품질 오디오 컨텐츠의 생성, 배포 및 재생을 용이하게 해준다.

<54> 본 명세서에 기술된 다양한 기법 및 도구가 독립적으로 사용될 수 있다. 이들 기법 및 도구 중 어떤 것은 조합하여(예를 들어, 결합된 인코딩 및/또는 디코딩 프로세스의 서로 다른 단계에서) 사용될 수 있다.

<55> 처리 동작들의 플로우차트를 참조하여 다양한 기법들이 이하에 기술되어 있다. 플로우차트에 도시된 다양한 처리 동작들은 더 적은 동작들로 통합되거나 더 많은 동작들로 분리될 수 있다. 간단함을 위해, 특정의 플로우차트에 도시된 동작들의 다른 곳에서 기술되는 동작들과의 관계가 종종 도시되어 있지 않다. 많은 경우에, 플로우차트에 있는 동작들은 재정렬될 수 있다.

<56> **I. 인코더 및/또는 디코더에 대한 예시적인 운영 환경**

<57> 도 1은 기술된 실시예들 중 몇가지가 구현될 수 있는 적당한 컴퓨팅 환경(100)의 일반화된 일례를 나타낸 것이다. 컴퓨팅 환경(100)은 용도 또는 기능성의 범위에 관한 어떤 제한을 암시하려는 것이 아닌데, 그 이유는 기술된 기법 및 도구가 다양한 범용 또는 특수 목적의 컴퓨팅 환경에서 구현될 수 있기 때문이다.

<58> 도 1을 참조하면, 컴퓨팅 환경(100)은 적어도 하나의 처리 장치(110) 및 메모리(120)를 포함한다. 도 1에서, 이러한 가장 기본적인 구성(130)은 점선 내부에 포함되어 있다. 처리 장치(110)는 컴퓨터 실행가능 명령어들을 실행하고 실제 프로세서(real processor) 또는 가상 프로세서(virtual processor)일 수 있다. 멀티-프로세싱

시스템에서는, 처리 능력을 향상시키기 위해 다수의 처리 장치가 컴퓨터 실행가능 명령어들을 실행한다. 메모리(120)는 휘발성 메모리(예를 들어, 레지스터, 캐쉬, RAM), 비휘발성 메모리(예를 들어, ROM, EEPROM, 플래쉬 메모리, 기타), 또는 이 둘의 어떤 조합일 수 있다. 메모리(120)는 본 명세서에 기술된 기법들 중 하나 이상을 사용하는 인코더 및/또는 디코더를 구현하는 소프트웨어(180)를 저장한다.

<59> 컴퓨팅 환경은 부가적인 특징들을 가질 수 있다. 예를 들어, 컴퓨팅 환경(100)은 저장 장치(140), 하나 이상의 입력 장치(150), 하나 이상의 출력 장치(160), 및 하나 이상의 통신 접속(170)을 포함한다. 버스, 컨트롤러, 또는 네트워크 등의 상호접속 메카니즘(도시 생략)은 컴퓨팅 환경(100)의 컴포넌트들을 상호접속시킨다. 일반적으로, 운영 체제 소프트웨어(도시 생략)는 컴퓨팅 환경(100)에서 실행되는 기타 소프트웨어에 대한 운영 환경을 제공하고 컴퓨팅 환경(100)의 컴포넌트들의 동작들을 조정한다.

<60> 저장 장치(140)는 이동식 또는 비이동식일 수 있으며, 자기 디스크, 자기 테이프 또는 카세트, CD-ROM, DVD, 또는 컴퓨팅 환경(100) 내에서 액세스될 수 있고 정보를 저장하는 데 사용될 수 있는 임의의 다른 매체를 포함한다. 저장 장치(140)는 소프트웨어(180)에 대한 명령어를 저장한다.

<61> 입력 장치(들)(150)는 키보드, 마우스, 펜 또는 트랙볼 등의 터치 입력 장치, 음성 입력 장치, 스캐닝 장치, 또는 컴퓨팅 환경(100)에 입력을 제공하는 다른 장치일 수 있다. 오디오 또는 비디오 인코딩의 경우, 입력 장치(들)(150)는 마이크, 사운드 카드, 비디오 카드, TV 튜너 카드, 또는 아날로그 또는 디지털 형태로 오디오 또는 비디오 입력을 받는 유사한 장치, 또는 오디오 또는 비디오 샘플을 컴퓨팅 환경(100)으로 읽어들이는 CD-ROM이나 CD-RW일 수 있다. 출력 장치(들)(160)는 디스플레이, 프린터, 스피커, CD-라이터, 또는 컴퓨팅 환경(100)으로부터의 출력을 제공하는 다른 장치일 수 있다.

<62> 통신 접속(들)(170)은 통신 매체를 통해 다른 컴퓨팅 개체로의 통신을 가능하게 해준다. 통신 매체는 컴퓨터 실행가능 명령어, 오디오 또는 비디오 입력 또는 출력, 또는 기타 데이터 등의 정보를 피변조 데이터 신호로 전달한다. 피변조 데이터 신호란 정보를 신호에 인코딩하도록 그 신호의 특성들 중 하나 이상이 설정 또는 변경된 신호를 말한다. 제한이 아닌 예로서, 통신 매체는 전기, 광학, RF, 적외선, 음향 또는 기타 반송파로 구현되는 유선 또는 무선 기법을 포함한다.

<63> 이들 기법 및 도구는 일반적으로 컴퓨터 판독가능 매체와 관련하여 기술될 수 있다. 컴퓨터 판독가능 매체는 컴퓨팅 환경 내에서 액세스될 수 있는 임의의 이용가능한 매체이다. 제한이 아닌 예로서, 컴퓨팅 환경(100)에서, 컴퓨터 판독가능 매체는 메모리(120), 저장 장치(140), 통신 매체, 및 상기한 것들 중 임의의 것의 조합을 포함한다.

<64> 이들 기법 및 도구는 일반적으로 컴퓨팅 환경에서 타겟 실제 또는 가상 프로세서 상에서 실행되는, 프로그램 모듈에 포함되어 있는 것 등의 컴퓨터 실행가능 명령어와 관련하여 기술될 수 있다. 일반적으로, 프로그램 모듈은 특정의 태스크를 수행하거나 특정의 추상 데이터 유형을 구현하는 루틴, 프로그램, 라이브러리, 객체, 클래스, 컴포넌트, 데이터 구조, 기타 등등을 포함한다. 프로그램 모듈의 기능은 다양한 실시예들에서 원하는 바에 따라 프로그램 모듈들 간에 결합 또는 분리될 수 있다. 프로그램 모듈에 대한 컴퓨터 실행가능 명령어는 로컬 또는 분산 컴퓨팅 환경 내에서 실행될 수 있다.

<65> 설명을 위해, 상세한 설명은 컴퓨팅 환경에서의 컴퓨터 동작을 기술하기 위해 "신호", "판정한다" 및 "적용한다"와 같은 용어를 사용한다. 이들 용어는 컴퓨터에 의해 수행되는 동작들에 대한 상위-레벨 추상화(high-level abstraction)이며, 사람에 의해 수행되는 동작들과 혼동해서는 안된다. 이들 용어에 대응하는 실제의 컴퓨터 동작은 구현에 따라 다르다.

<66> II. 예시적인 인코더 및 디코더

<67> 도 2는 하나 이상의 기술된 실시예들이 구현될 수 있는 제1 오디오 인코더(200)를 도시한 것이다. 인코더(200)는 변환-기반의 지각 오디오 인코더(200)이다. 도 3은 대응하는 오디오 디코더(300)를 도시한 것이다.

<68> 도 4는 하나 이상의 기술된 실시예들이 구현될 수 있는 제2 오디오 인코더(400)를 도시한 것이다. 인코더(400)는, 다시 말하자면, 변환-기반의 지각 오디오 인코더이지만, 인코더(400)는 다중-채널 오디오를 처리하기 위한 부가적인 모듈들을 포함하고 있다. 도 5는 대응하는 오디오 디코더(500)를 도시한 것이다.

<69> 도 6은 하나 이상의 기술된 실시예들이 구현될 수 있는 더 일반화된 미디어 인코더(600)를 도시한 것이다. 도 7은 대응하는 미디어 디코더(700)를 도시한 것이다.

<70> 도 2 내지 도 7에 도시된 시스템이 일반화되어 있지만, 각각은 실세계 시스템에서 발견되는 특징들을 갖는다.

어쨌든, 인코더 및 디코더 내의 모듈들 간에 도시된 관계들은 인코더 및 디코더에서의 정보의 흐름을 나타낸 것이며, 간단함을 위해 다른 관계들은 도시되어 있지 않다. 구현 및 원하는 압축 유형에 따라, 인코더 또는 디코더의 모듈들이 추가, 생략, 다수의 모듈로 분할, 다른 모듈들과 결합, 및/또는 유사한 모듈로 대체될 수 있다. 대안의 실시예들에서, 다른 모듈 및/또는 기타 구성을 갖는 인코더 또는 디코더가 하나 이상의 기술된 실시예들에 따라 오디오 데이터 또는 어떤 다른 유형의 데이터를 처리한다. 예를 들어, 스펙트럼 계수를 처리하는 도 2 내지 도 7에서의 모듈들은 기저 대역(base band) 또는 기본 주파수 서브-영역(들)(base frequency sub-range)(하위 주파수 등)에서의 계수들만을 처리하는 데 사용될 수 있으며, 다른 모듈들(도시 생략)은 다른 주파수 서브-영역들(상위 주파수 등)에서의 스펙트럼 계수를 처리한다.

<71> **A. 제1 오디오 인코더**

<72> 일반적으로, 인코더(200)는 어떤 샘플링 깊이 및 레이트로 입력 오디오 샘플(205)의 시계열(time series)을 수신한다. 입력 오디오 샘플(205)은 다중-채널 오디오(예를 들어, 스테레오) 또는 모노 오디오에 대한 것이다. 인코더(200)는 오디오 샘플(205)을 압축하고 인코더(200)의 여러가지 모듈에 의해 생성된 정보를 멀티플렉싱하여 WMA 형식, ASF(Advanced Streaming Format), 또는 기타 형식 등의 형식으로 비트스트림(295)을 출력한다.

<73> 주파수 변환기(210)는 오디오 샘플(205)을 수신하고 이를 주파수 영역의 데이터로 변환한다. 예를 들어, 주파수 변환기(210)는 오디오 샘플(205)을, 가변 시간 해상도(variable temporal resolution)를 가능하게 해주기 위해 가변 크기를 가질 수 있는 블록들로 분할한다. 그렇지 않으면 나중의 양자화에 의해 유발될 수 있는 블록들 간의 지각가능한 불연속을 감소시키기 위해 블록들이 중첩할 수 있다. 주파수 변환기(210)는 시변 MLT(time-varying Modulated Lapped Transform), MDCT(modulated DCT), MLT나 DCT의 어떤 다른 변형, 또는 어떤 다른 유형의 변조 또는 비변조, 중첩 또는 비중첩 주파수 변환을 블록들에 적용하거나, 또는 서브대역 또는 웨이블릿 코딩을 사용한다. 주파수 변환기(210)는 스펙트럼 계수 데이터의 블록들을 출력하고 블록 크기 등의 부수 정보를 멀티플렉서(MUX)(280)로 출력한다.

<74> 다중-채널 오디오 데이터의 경우, 다중-채널 변환기(220)는 다수의 원래의 독립적으로 코딩된 채널을 결합 코딩된 채널로 변환할 수 있다. 또는, 다중-채널 변환기(220)는 좌채널 및 우채널을 독립적으로 코딩된 채널로서 통과시킬 수 있다. 다중-채널 변환기(220)는 사용되는 채널 모드를 가리키는 부수 정보를 생성하여 MUX(280)로 보낸다. 인코더(200)는 다중-채널 변환 이후에 오디오 데이터 블록에 다중-채널 리매트릭싱(multi-channel rematrixing)을 적용할 수 있다.

<75> 지각 모델러(perception modeler)(230)는 주어진 비트 레이트에 대해 재구성된 오디오 신호의 지각 품질을 향상시키기 위해 사람의 청각 시스템의 특성을 모델링한다. 지각 모델러(230)는 다양한 청각 모델 중 임의의 것을 사용한다.

<76> 지각 모델러(230)는 가중기(weighter)(240)가 노이즈의 가청도를 감소시키기 위해 오디오 데이터 내의 노이즈를 쉐이핑하는 데 사용하는 정보를 출력한다. 예를 들어, 다양한 기법들 중 임의의 것을 사용하여, 가중기(240)는 양자화 행렬(때때로 마스크(mask)라고 함)에 대한 가중 인자(때때로 스케일 인자라고 함)를 발생한다. 가중기(240)는 이어서 다중-채널 변환기(220)로부터 수신되는 데이터에 가중 인자를 적용한다. 일련의 가중 인자가 더 효율적인 표현을 위해 압축될 수 있다.

<77> 양자화기(250)는 가중기(240)의 출력을 양자화하고, 양자화된 계수 데이터를 생성하여 엔트로피 인코더(260)로 보내고 양자화 스텝 크기를 비롯한 부수 정보를 MUX(280)로 보낸다. 도 2에서, 양자화기(250)는 적응적, 균일, 스칼라 양자화기이다. 양자화기(250)는 각각의 스펙트럼 계수에 동일한 양자화 스텝 크기를 적용하지만, 이 양자화 스텝 크기 자체는 엔트로피 인코더(260) 출력의 비트 레이트에 영향을 주기 위해 양자화 루프의 반복마다 변할 수 있다. 다른 종류의 양자화는 비균일, 벡터 양자화 및/또는 비적응적 양자화이다.

<78> 엔트로피 인코더(260)는 양자화기(250)로부터 수신되는 양자화된 계수 데이터를 무손실 압축한다, 예를 들어, 런-레벨 코딩 및 벡터 가변 길이 코딩을 수행한다. 어떤 실시예들에서의 다양한 엔트로피 인코딩(아마도 전처리(preprocessing)를 포함함) 메카니즘이 섹션 III 내지 V에서 상세히 기술된다. 다른 대안으로서, 엔트로피 인코더(260)는 어떤 다른 형태 또는 조합의 엔트로피 코딩 메카니즘을 사용한다. 엔트로피 인코더(260)는 오디오 정보를 인코딩하는 데 소비되는 비트수를 계산하고 이 정보를 레이트/품질 제어기(270)로 전달할 수 있다.

<79> 제어기(270)는 인코더(200)의 출력의 비트 레이트 및/또는 품질을 조절하기 위해 양자화기(250)에 작용한다. 제어기(270)는 비트 레이트 및 품질 제약조건을 만족시키기 위해 양자화 스텝 크기를 양자화기(250)로 출력한다.

<80> 그에 부가하여, 인코더(200)는 오디오 데이터 블록에 노이즈 대체(noise substitution) 및/또는 대역 절단(band truncation)을 적용할 수 있다.

<81> MUX(280)는 오디오 인코더(200)의 다른 모듈들로부터 수신되는 부수 정보를, 엔트로피 인코더(260)로부터 수신되는 엔트로피 인코딩된 데이터와 함께, 멀티플렉싱한다. MUX(280)는 인코더(200)에 의해 출력되는 비트스트림(295)을 저장하는 가상 버퍼를 포함할 수 있다.

<82> **B. 제1 오디오 디코더**

<83> 일반적으로, 디코더(300)는 엔트로피 인코딩된 데이터는 물론 부수 정보를 비롯한 압축된 오디오 정보의 비트스트림(305)을 수신하며, 이 비트스트림(305)으로부터 디코더(300)는 오디오 샘플(395)을 재구성한다.

<84> 디멀티플렉서(DEMUX)(310)는 비트스트림(305) 내의 정보를 파싱하고 정보를 디코더(300)의 모듈들로 전송한다. DEMUX(310)는 오디오의 복잡도의 변동, 네트워크 지터 및/또는 기타 인자로 인한 비트 레이트의 단기 변화를 보상하기 위해 하나 이상의 버퍼를 포함한다.

<85> 엔트로피 디코더(320)는 DEMUX(310)로부터 수신되는 엔트로피 코드를 무손실 압축해제하여, 양자화된 스펙트럼 계수 데이터를 생성한다. 엔트로피 디코더(320)는 일반적으로 인코더에서 사용되는 엔트로피 인코딩 기법의 반대(inverse)를 적용한다. 어떤 실시예들에서의 다양한 엔트로피 디코딩 메카니즘이 섹션 III 내지 V에서 상세히 기술된다.

<86> 역양자화기(330)는 DEMUX(310)로부터 양자화 스텝 크기를 수신하고 엔트로피 디코더(320)로부터 양자화된 스펙트럼 계수 데이터를 수신한다. 역양자화기(330)는 주파수 계수 데이터를 부분적으로 재구성하기 위해 양자화 스텝 크기를 양자화된 주파수 계수 데이터에 적용하거나, 그렇지 않고 역양자화를 수행한다.

<87> DEMUX(310)로부터, 노이즈 발생기(340)는 데이터 블록 내의 어느 대역이 노이즈 대체되는지는 물론 노이즈의 형태에 대한 임의의 파라미터를 나타내는 정보를 수신한다. 노이즈 발생기(340)는 표시된 대역에 대한 패턴을 발생하고 이 정보를 역가중기(350)로 전달한다.

<88> 역가중기(350)는 DEMUX(310)로부터 가중 인자를 수신하고, 노이즈 발생기(340)로부터 임의의 노이즈-대체된 대역에 대한 패턴을 수신하며, 역양자화기(330)로부터 부분적으로 재구성된 주파수 계수 데이터를 수신한다. 필요에 따라, 역가중기(350)는 가중 인자를 압축해제한다. 역가중기(350)는 노이즈 대체되지 않은 대역에 대한 부분적으로 재구성된 주파수 계수 데이터에 가중 인자를 적용한다. 역가중기(350)는 이어서 노이즈-대체된 대역에 대해 노이즈 발생기(340)로부터 수신된 노이즈 패턴을 부가한다.

<89> 역다중-채널 변환기(360)는 역가중기(350)로부터 재구성된 스펙트럼 계수 데이터를 수신하고 DEMUX(310)로부터 채널 모드 정보를 수신한다. 다중-채널 오디오가 독립적으로 코딩된 채널에 있는 경우, 역다중-채널 변환기(360)는 그 채널들을 통과시킨다. 다중-채널 데이터가 결합 코딩된 채널에 있는 경우, 역다중-채널 변환기(360)는 그 데이터를 독립적으로 코딩된 채널로 변환한다.

<90> 역주파수 변환기(370)는 다중-채널 변환기(360)에 의해 출력되는 스펙트럼 계수 데이터는 물론 DEMUX(310)로부터의 블록 크기 등의 부수 정보도 수신한다. 역주파수 변환기(370)는 인코더에서 사용되는 주파수 변환의 반대를 적용하고 재구성된 오디오 샘플(395)의 블록을 출력한다.

<91> **C. 제2 오디오 인코더**

<92> 도 4를 참조하면, 인코더(400)는 어떤 샘플링 깊이 및 레이트로 입력 오디오 샘플(405)의 시계열을 수신한다. 입력 오디오 샘플(405)은 다중-채널 오디오(예를 들어, 스테레오, 서라운드) 또는 모노 오디오에 대한 것이다. 인코더(400)는 오디오 샘플(405)을 압축하고 인코더(400)의 다양한 모듈들에 의해 생성되는 정보를 멀티플렉싱하여 WMA Pro 형식 또는 기타 형식 등의 형식으로 비트스트림(495)을 출력한다.

<93> 인코더(400)는 오디오 샘플(405)에 대한 다수의 인코딩 모드 중에서 선택한다. 도 4에서, 인코더(400)는 혼합/순수(mixed/pure) 무손실 코딩 모드 및 손실 코딩 모드 간에 전환한다. 무손실 코딩 모드는 혼합/순수 무손실 코더(472)를 포함하고 일반적으로 고품질(및 고 비트 레이트(high bit rate)) 압축을 위해 사용된다. 손실 코딩 모드는 가중기(442) 및 양자화기(460) 등의 컴포넌트들을 포함하고 일반적으로 조정가능한(및 제어 비트 레이트(controlled bit rate)) 품질 압축을 위해 사용된다. 선택 결정은 사용자 입력 또는 다른 기준에 달려 있다.

<94> 다중-채널 오디오 데이터의 손실 코딩의 경우, 다중-채널 프리-프로세서(multi-channel pre-processor)(410)는

선택에 따라서는 시간-영역 오디오 샘플(405)을 리매트릭싱한다. 어떤 실시예들에서, 다중-채널 프리-프로세서(410)는 오디오 샘플(405)을 선택적으로 리매트릭싱하여 하나 이상의 코딩된 채널을 드롭(drop)시키거나 인코더(400)에서 채널간 상관(inter-channel correlation)을 증가시키지만, 디코더(500)에서 (어떤 형태로) 재구성을 가능하게 해준다. 다중-채널 프리-프로세서(410)는 다중-채널 후처리(multi-channel post-processing)를 위한 명령어 등의 부수 정보를 MUX(490)로 전송할 수 있다.

<95> 윈도잉 모듈(windowing module)(420)은 한 프레임의 오디오 입력 샘플(405)을 서브-프레임 블록(윈도우)으로 분할한다. 이 윈도우는 시변 크기(time-varying size) 및 윈도우 쉐이핑 함수(window shaping function)를 가질 수 있다. 인코더(400)가 손실 코딩을 사용할 때, 가변-크기 윈도우가 가변 시간 해상도(variable temporal resolution)를 가능하게 해준다. 윈도잉 모듈(420)은 분할된 데이터의 블록을 출력하고 블록 크기 등의 부수 정보를 MUX(490)로 출력한다.

<96> 도 4에서, 타일 구성기(tile configurer)(422)는 다중-채널 오디오의 프레임을 채널별로 분할한다. 품질/비트 레이트가 허용하는 경우, 타일 구성기(422)는 프레임 내의 각각의 채널을 독립적으로 분할한다. 예를 들어, 타일 구성기(422)는 시간상으로 같은 장소에 있는 동일한 크기의 윈도우를 타일로서 그룹화한다.

<97> 주파수 변환기(430)는 오디오 샘플을 수신하고 이들을 주파수 영역의 데이터로 변환하여, 도 2의 주파수 변환기(210)에 대해 상기한 것 등의 변환을 적용한다. 주파수 변환기(430)는 스펙트럼 계수 데이터의 블록을 가중기(442)로 출력하고 블록 크기 등의 부수 정보를 MUX(490)로 출력한다. 주파수 변환기(430)는 주파수 계수 및 부수 정보 둘다를 지각 모델러(440)로 출력한다.

<98> 지각 모델러(440)는 사람의 청각 시스템의 특성을 모델링하고, 청각 모델에 따라 오디오 데이터를 처리한다.

<99> 가중기(442)는 지각 모델러(440)로부터 수신되는 정보에 기초하여 양자화 행렬에 대한 가중 인자를 발생한다. 가중기(442)는 주파수 변환기(430)로부터 수신되는 데이터에 가중 인자를 적용한다. 가중기(442)는 양자화 행렬 및 채널 가중 인자 등의 부수 정보를 MUX(490)로 출력하고, 양자화 행렬이 압축될 수 있다.

<100> 다중-채널 오디오 데이터의 경우, 다중-채널 변환기(450)가 다중-채널 변환을 적용할 수 있다. 예를 들어, 다중-채널 변환기(450)는 선택적으로 또 융통성있게 타일 내의 채널 및/또는 양자화 대역의 전부가 아닌 일부에 다중-채널 변환을 적용한다. 다중-채널 변환기(450)는 사전-정의된 행렬 또는 커스텀 행렬(custom matrix)을 선택적으로 사용하고, 커스텀 행렬에 효율적인 압축을 적용한다. 다중-채널 변환기(450)는, 예를 들어, 사용되는 다중-채널 변환 및 타일의 다중-채널 변환된 부분을 나타내는 부수 정보를 생성하여 MUX(490)로 보낸다.

<101> 양자화기(460)는 다중-채널 변환기(450)의 출력을 양자화하고, 양자화된 계수 데이터를 생성하여 엔트로피 인코더(470)로 보내고 양자화 스텝 크기를 비롯한 부수 정보를 MUX(490)로 보낸다. 도 4에서, 양자화기(460)는 타일마다 양자화 인자를 계산하는 적응적, 균일, 스칼라 양자화기이지만, 양자화기(460)는 그 대신에 어떤 다른 종류의 양자화를 수행할 수 있다.

<102> 엔트로피 인코더(470)는, 도 2의 엔트로피 인코더(260)를 참조하여 전체적으로 상기한 바와 같이, 양자화기(460)로부터 수신되는 양자화된 계수 데이터를 무손실 압축한다. 몇몇 실시예들에서의 다양한 엔트로피 인코딩 메카니즘(아마도 전처리를 포함함)이 섹션 III 내지 V에서 상세히 기술된다.

<103> 제어기(480)는 인코더(400)의 출력의 비트 레이트 및/또는 품질을 조절하기 위해 양자화기(460)에 작용한다. 제어기(480)는 품질 및/또는 비트 레이트 제약조건을 만족시키기 위해 양자화기(460)로 양자화 인자를 출력한다.

<104> 혼합/순수 무손실 인코더(472) 및 연관된 엔트로피 인코더(474)는 혼합/순수 무손실 코딩 모드에 대한 오디오 데이터를 압축한다. 인코더(400)는 전체 시퀀스에 대해 혼합/순수 무손실 코딩 모드를 사용하거나, 프레임별로, 블록별로, 타일별로, 또는 다른 방식으로 코딩 모드 간에 전환한다.

<105> MUX(490)는 오디오 인코더(400)의 다른 모듈들로부터 수신되는 부수 정보를, 엔트로피 인코더(470, 474)로부터 수신되는 엔트로피 인코딩된 데이터와 함께, 멀티플렉싱한다. MUX(490)는 레이트 제어 또는 다른 목적을 위해 하나 이상의 버퍼를 포함한다.

<106> **D. 제2 오디오 디코더**

<107> 도 5를 참조하면, 제2 오디오 디코더(500)는 압축된 오디오 정보의 비트스트림(505)을 수신한다. 비트스트림(505)은 엔트로피 인코딩된 데이터는 물론 부수 정보(이들로부터 디코더(500)는 오디오 샘플(595)을 재구성함)

도 포함한다.

<108> DEMUX(510)는 비트스트림(505) 내의 정보를 파싱하고 정보를 디코더(500)의 모듈들로 전송한다. DEMUX(510)는 오디오의 복잡도의 변동, 네트워크 지터, 및/또는 다른 인자들로 인한 비트 레이트의 단기 변화를 보상하기 위해 하나 이상의 버퍼를 포함한다.

<109> 엔트로피 디코더(520)는 DEMUX(510)로부터 수신되는 엔트로피 코드를 무손실 압축해제하고, 일반적으로 인코더(400)에서 사용되는 엔트로피 인코딩 기법의 반대를 적용한다. 손실 코딩 모드로 압축된 데이터를 디코딩할 때, 엔트로피 디코더(520)는 양자화된 스펙트럼 계수 데이터를 생성한다. 어떤 실시예들에서의 다양한 엔트로피 디코딩 메카니즘이 섹션 III 내지 V에서 상세히 기술된다.

<110> 혼합/순수 무손실 디코더(522) 및 연관된 엔트로피 디코더(들)(520)는 혼합/순수 무손실 코딩 모드에 대한 무손실 인코딩된 오디오 데이터를 압축해제한다.

<111> 타일 구성 디코더(530)는 DEMUX(510)로부터 프레임들에 대한 타일의 패턴을 나타내는 정보를 수신하고, 필요한 경우, 이를 디코딩한다. 타일 패턴 정보는 엔트로피 인코딩되거나 다른 방식으로 파라미터화(parameterize)될 수 있다. 타일 구성 디코더(530)는 이어서 타일 패턴 정보를 디코더(500)의 다양한 다른 모듈들로 전달한다.

<112> 역다중-채널 변환기(540)는 엔트로피 디코더(520)로부터의 양자화된 스펙트럼 계수 데이터는 물론, 타일 구성 디코더(530)로부터의 타일 패턴 정보 및, 예를 들어, 사용되는 다중-채널 변환 및 타일의 변환된 부분을 나타내는 DEMUX(510)로부터의 부수 정보도 수신한다. 이러한 정보를 사용하여, 역다중-채널 변환기(540)는 필요에 따라 변환 행렬을 압축해제하고, 선택적으로 또 융통성있게 하나 이상의 역다중-채널 변환을 오디오 데이터에 적용한다.

<113> 역양자화기/가중기(550)는 DEMUX(510)로부터 타일 및 채널 양자화 인자는 물론 양자화 행렬도 수신하고, 역다중-채널 변환기(540)로부터 양자화된 스펙트럼 계수 데이터를 수신한다. 역양자화기/가중기(550)는, 필요에 따라, 수신된 양자화 인자/행렬 정보를 압축해제하고, 이어서 역양자화 및 가중을 수행한다.

<114> 역주파수 변환기(560)는 역양자화기/가중기(550)에 의해 출력되는 스펙트럼 계수 데이터는 물론 DEMUX(510)로부터의 부수 정보 및 타일 구성 디코더(530)로부터의 타일 패턴 정보도 수신한다. 역주파수 변환기(570)는 인코더에서 사용되는 주파수 변환의 반대를 적용하고 블록들을 중첩기/가산기(overlapper/adder)(570)로 출력한다.

<115> 타일 구성 디코더(530)로부터 타일 패턴 정보를 수신하는 것에 부가하여, 중첩기/가산기(570)는 역주파수 변환기(560) 및/또는 혼합/순수 무손실 디코더(522)로부터 디코딩된 정보를 수신한다. 중첩기/가산기(570)는, 필요에 따라, 오디오 데이터를 중첩 및 가산하고 인코딩된 오디오 데이터의 프레임 또는 기타 시퀀스를 다른 모드와 인터리빙(interleave)한다.

<116> 다중-채널 포스트-프로세서(multi-channel post-processor)(580)는 선택에 따라서 중첩기/가산기(570)에 의해 출력되는 시간-영역 오디오 샘플을 리매트릭싱한다. 비트스트림-제어 후처리(bitstream-controlled post-processing)의 경우, 후처리 변환 행렬이 시간에 따라 변하며 비트스트림(505)으로 신호되거나 그에 포함된다.

<117> **E. 일반화된 미디어 인코더**

<118> 도 6은 오디오, 비디오 또는 기타 미디어 컨텐츠를 인코딩하는 일반화된 미디어 인코더(600)의 일부를 나타낸 것이다. 간단함을 위해, 인코더(600)의 수많은 모듈들 및 여러 유형의 부수 정보(미디어 컨텐츠의 유형에 따라 다를 수 있음)가 도시되어 있지 않다.

<119> 도 2 및 도 4에 각각 도시된 인코더(200, 400)와 같이, 도 6에 도시된 입력이 양자화되지 않은 스펙트럼 계수(605)인 한, 인코더(600)는 변환-기반이다. 그렇지만, 어떤 실시예들에서, 본 명세서에 기술된 엔트로피 인코딩 메카니즘 중 하나(예를 들어, 섹션 V에 기술된 메카니즘)가 어떤 다른 종류의 입력에 대해 수행된다.

<120> 양자화기(620)는 계수(605)를 양자화하여, 양자화된 계수 데이터를 생성한다. 예를 들어, 양자화기(620)는 적응적, 균일, 스칼라 양자화기 또는 어떤 다른 종류의 양자화기이다.

<121> 엔트로피 코딩 프로프로세서(604)는 엔트로피 코딩 이전에 선택적으로 전처리를 수행한다. 예를 들어, 프로프로세서(640)는, 섹션 III에 기술된 바와 같이, 양자화된 스펙트럼 계수에 대해 계수 예측을 수행한다. 또는, 프리프로세서(640)는, 섹션 IV에 기술된 바와 같이, 양자화된 스펙트럼 계수를 재정렬한다. 다른 대안으로서, 프리프로세서(640)는 어떤 다른 유형의 전처리를 수행한다.

<122> 전처리된 계수는 별도로 하고, 프리프로세서(640)는 전처리를 기술하는 부수 정보를 출력 비트스트림(695)으로 출력한다. 예를 들어, 부수 정보는, 섹션 III에서 기술되는 바와 같이, 계수 예측에서 사용되는 예측 인자를 포함한다. 또는, 부수 정보는, 섹션 IV에서 기술된 바와 같이, 양자화된 스펙트럼 계수를 재정렬하는 데 사용되는 정보를 포함한다.

<123> 엔트로피 인코더(660)는 양자화된 계수 데이터를 무손실 압축하고, 예를 들어, 런-레벨 코딩(run-level coding) 및 벡터 가변 길이 코딩(vector variable length coding)을 수행한다. 섹션 V는 적응적 엔트로피 인코딩 메카니즘에 대해 기술한다. 다른 대안으로서, 엔트로피 인코더(660)는 어떤 다른 형태 또는 조합의 엔트로피 코딩 메카니즘을 사용한다.

<124> 도 6이 단지 엔트로피 인코더(660)로부터의 피드백 없이 엔트로피 인코더(660)에 입력을 제공하고 전처리를 수행하는 프리프로세서(640)만을 도시하고 있지만, 다른 대안으로서, 엔트로피 인코더(660)는 프리프로세서(640)에 피드백을 제공하고, 프리프로세서(640)는 이 피드백을 사용하여 전처리를 조정한다. 예를 들어, 프리프로세서(640)는, 엔트로피 인코더(660)가 엔트로피 인코딩 모델에 더 잘 맞도록, 엔트로피 인코더(660)로부터의 피드백에 기초하여 계수 재정렬을 조정한다.

<125> **F. 일반화된 미디어 디코더**

<126> 도 7은 오디오, 비디오, 또는 기타 미디어 컨텐츠를 디코딩하는 일반화된 미디어 디코더(700)의 일부를 나타낸 것이다. 간단함을 위해, 디코더(700)의 수많은 모듈들 및 여러 유형의 부수 정보(미디어 컨텐츠의 유형에 따라 다를 수 있음)가 도시되어 있지 않다.

<127> 도 3 및 도 5에 각각 도시된 디코더(300, 500)와 같이, 도 7에 나타낸 출력이 재구성된 스펙트럼 계수(705)인 한, 디코더(700)는 변환-기반이다. 그렇지만, 몇몇 실시예들에서, 본 명세서에 기술된 엔트로피 디코딩 메카니즘들 중 하나가 어떤 다른 종류의 출력에 대해 수행된다.

<128> 엔트로피 디코더(760)는 양자화된 계수 데이터를 무손실 압축해제하고, 예를 들어, 런-레벨 디코딩 및 벡터 가변 길이 디코딩을 수행한다. 섹션 V는 적응적 엔트로피 디코딩 메카니즘에 대해 기술한다. 다른 대안으로서, 엔트로피 디코더(760)는 어떤 다른 형태 또는 조합의 엔트로피 디코딩 메카니즘을 사용한다.

<129> 엔트로피 디코딩 포스트프로세서(740)는 엔트로피 디코딩 이후에 선택적으로 후처리를 수행한다. 예를 들어, 포스트프로세서(740)는, 섹션 III에 기술된 바와 같이, 양자화된 스펙트럼 계수에 대해 계수 예측을 수행한다. 또는, 포스트프로세서(740)는, 섹션 IV에 기술된 바와 같이, 양자화된 스펙트럼 계수를 재정렬한다. 다른 대안으로서, 포스트프로세서(740)는 어떤 다른 유형의 후처리를 수행한다.

<130> 엔트로피 디코딩된 계수는 별도로 하고, 포스트프로세서(740)는 후처리에 대해 기술하는 부수 정보를 비트스트림(795)으로부터 수신한다. 예를 들어, 이 부수 정보는, 섹션 III에 기술된 바와 같이, 계수 예측에서 사용되는 예측 인자를 포함한다. 또는, 이 부수 정보는, 섹션 IV에 기술된 바와 같이, 양자화된 스펙트럼 계수를 재정렬하는 데 사용되는 정보를 포함한다.

<131> 역양자화기(720)는 역양자화를 수행하여, 재구성된 계수(705) 데이터를 생성한다. 예를 들어, 역양자화기(720)는 적응적, 균일, 스칼라 역양자화기 또는 어떤 다른 종류의 양자화기이다.

<132> **III. 코딩 및 디코딩을 위한 스펙트럼 영역에서의 계수의 예측**

<133> 오디오 인코더는 종종 변환 코딩과 그에 뒤이어서 양자화 및 엔트로피 코딩을 사용하여 압축을 달성한다. 고정된 변환이 사용될 때, 오디오 신호의 어떤 패턴의 경우, 변환 이후에 인접한 계수들 간에 여전히 상관(correlation)이 있다. 코딩 효율을 향상시키기 위해 이러한 상관을 이용하는 다양한 기법 및 도구에 대해 이하에서 기술한다. 상세하게는, 어떤 실시예들에서, 도 2, 도 4 또는 도 6에 도시된 것과 같은 인코더는 인코딩 동안에 양자화된 스펙트럼 계수에 대해 계수 예측을 수행한다. (도 3, 도 5 또는 도 7에 도시된 것과 같은) 대응하는 디코더는 디코딩 동안에 양자화된 스펙트럼 계수에 대해 계수 예측을 수행한다.

<134> **A. 예시적인 문제 영역**

<135> 파형인 오디오를 압축하는 일반적인 오디오 인코더에서, 입력 오디오 신호는 가변 윈도우 크기 MDCT 또는 가변-크기 윈도우를 갖는 다른 변환을 사용하여 변환된다. 예를 들어, 도 8a에 도시된 스테레오 오디오의 윈도잉 분석(windowing analysis)의 결과, 도 8b에 도시된 윈도우 구성이 얻어지는 것으로 가정하자. 일반적으로, 이러한 윈도우 구성은 (다른 세그먼트에 대해 더 긴 윈도우를 사용함으로써) 전체적인 코딩 효율을 용이하게 해주면

서 (과도 세그먼트(transient segment)에 대해 더 짧은 윈도우를 사용함으로써) 디코딩된 신호에서 프리-에코(pre-echo) 및 포스트-에코(post-echo)를 감소시킨다. 윈도잉 분석의 한가지 목적은 임의의 주어진 윈도우 내의 신호가 대체로 정상 상태(stationary)인 윈도우 경계를 식별하는 것이다.

<136> 스펙트럼 계수는, 채널 변환 이전 또는 이후에, 양자화된다. 종래에, 서브-프레임 또는 기타 윈도우의 스펙트럼 계수는 그들 간에 어떤 선형 상관도 갖지 않는 것으로 가정된다. 오히려, 스펙트럼 계수가 보통 어떤 고차의 통계적 관계(인코더가 엔트로피 인코딩 동안에 이를 이용하려고 시도함)를 갖는 것으로 가정된다.

<137> 실제로, 이러한 인코딩에서 암시적인 몇가지 가정은 다양한 환경에서 적용되지 않는다. 예를 들어, 어떤 유형 및 패턴의 오디오 신호의 경우, 서브-프레임 또는 다른 윈도우에 대한 스펙트럼 계수들이 반드시 상관되지 않아야 하는 것은 아니다. 윈도우 내의 신호가 비정상 상태(non-stationary)일 수 있는 많은 같은 이유로 인해(이하 참조), 스펙트럼 계수는 선형 상관을 나타낼 수 있다. 최근의 파형-기반 인코더는 엔트로피 인코딩에서 이러한 상관을 이용하지 못한다.

<138> 다른 예로서, 윈도잉 분석이 어떤 오디오 신호에 적용될 때, 특정의 윈도우 내의 신호가 반드시 정상 상태이어야 하는 것은 아니다. 입력 오디오가 시간에 따라 심하게 변할 때(예를 들어, 음성 신호의 경우), 짧은 윈도우조차도 과도 세그먼트를 격리시키기에 충분하지 않을 수 있다. 또는, 레이트 제어기 내의 버퍼가 충만되어 있는 경우, 그렇지 않았으면 더 작은 윈도우가 사용될지라도, 제어기는 강제로 인코더에게 더 큰 윈도우를 사용하여 비트 레이트를 감소시키게 할 수 있다. 또는, 천이(transition)가 느린 경우, 윈도잉 분석이 과도 응답(transient)을 검출하지 못할 수 있으며, 따라서 더 짧은 윈도우가 도입되지 않는다. 또는, 윈도잉 분석은 프레임 내의 다른 과도응답들이 아니라 프레임마다 단지 하나의 과도응답에 의해 유입되는 프리-에코에 대해 보호할 수 있다. 또는, 윈도우 내의 신호가 어떤 다른 이유로 비정상 상태일 수 있다.

<139> 스케일 인자는 왜곡(distortion)의 스펙트럼 분포를 제어하는 데 도움을 줄 수 있다. 그렇지만, 왜곡의 시간 분포에 관해서는, 스펙트럼에 걸친 간단한 양자화가 전체 변환 블록에 걸쳐 일정한 왜곡을 유입하며, 이 왜곡이 프레임의 시간 세그먼트들에 가청 왜곡(audible distortion)을 야기할 수 있다.

<140> TNS(Temporal Noise Shaping)는 시간에 따라 양자화 노이즈를 쉐이핑하기 위해 주파수 영역에서 예측 방법을 사용하는 어떤 MPEG 변형에서의 기술이다. TNS에서, 전체 시간 윈도우에 걸쳐 양자화 노이즈가 스미어링(smearing)하는 것을 제한하기 위해, 인코더는 스펙트럼 계수에 예측 필터를 적용하고 필터링된 신호를 양자화한다. 도 9 및 도 10은 각각 인코더 및 디코더에서의 TNS를 나타낸 것이다.

<141> 도 9를 참조하면, 인코더는 양자화되지 않은 스펙트럼 계수(905)와 예측자(predictor)(2개의 이전의 재구성된 계수의 합성) 간의 차이를 계산한다. 이 합성(combination)을 위해, 2개의 재구성된, (지연(910, 912)에서의) 시간-지연된 계수는 각각 예측 인자(911, 913)와 곱해지고 서로 가산된다. 예측 인자(911, 913)는 양자화되고 비트스트림(995)에 포함된다. 양자화기(970)는 차이값을 양자화하고, 엔트로피 인코더(990)는 양자화된 차이값을 엔트로피 인코딩하여 비트스트림(995)으로 출력한다. 역양자화기(980)는 차이값을 재구성하고 이를 계수(905)에 대한 예측자에 가산한다. 이 결과, 계수가 재구성되고, 이는 제1 지연(910)에, 이어서 제2 지연(912)에 버퍼링되어 차후의 계수(905)에 대한 예측자에 기여하게 된다.

<142> 대응하는 디코더에서, 엔트로피 디코더(1090)는 비트스트림(1095)으로부터의 차이값을 엔트로피 디코딩하고, 역양자화기(1080)는 이 차이값을 역양자화한다. 디코더는 차이값을 예측자와 합성하여 재구성된 스펙트럼 계수(1005)를 생성하며, 여기서 예측자는 2개의 이전의 재구성된 계수의 합성이다. 이 합성의 계산은 2개의 지연(1010, 1012) 및 2개의 예측 인자(1011, 1013)(비트스트림(1095)으로부터 복원됨)를 필요로 한다. 재구성된 스펙트럼 계수(1005)는 제1 지연(1010)에, 이어서 제2 지연(1012)에 버퍼링되어 차후의 계수(1005)에 대한 예측자에 기여하게 된다.

<143> MPEG AAC에서의 TNS는 입력 신호의 서로 다른 스펙트럼 영역에 적용될 3개까지의 서로 다른 무한 임펄스 응답(infinite-impulse-response) 필터(또는 예측자)를 고려하고 있다. 필터 계수가 양자화되어 비트스트림에 포함된다.

<144> MPEG AAC가 짧은 윈도우(short window)의 사용을 허용할 때에도, TNS는 짧은 윈도우에서 사용되지 않는데, 그 이유는 예측자 기술 정보(predictor description information)에 대해 요구되는 전체 정보가 비교적 크며, 그 결과 스펙트럼 값에 대한 비트가 감소되기 때문이다. 그 자체로서, TNS는 MPEG AAC에서 긴 윈도우에 대해서만 고려되며, 이는 TNS의 효용성을 제한한다.

<145> 또한, 도 9 및 도 10에 도시한 바와 같이, TNS에서의 예측은 양자화되지 않은/재구성된 영역에서 일어난다. 그

결과, 디코더는 역양자화 및 예측(그리고, 아마도 심지어 엔트로피 디코딩)의 동작들을 인터리빙해야만 하며, 따라서 복잡도가 증가한다. 그에 부가하여, 양자화되지 않은/재구성된 영역에서의 예측의 경우, TNS 동작은 MPEG AAC에서 부동 소수점 연산으로서 규정되어 있으며, 이는 고정 소수점 구현에서 어려움을 야기한다.

<146> TNS 예측자는 2차 예측자이며, 각각의 스펙트럼 계수에서 예측 동작을 위해 2번의 곱셈을 필요로 한다. 인코더 측에서는, 효과적인 예측자의 설계가 어려울 수 있으며, 불안정한 예측자가 문제될 수 있다.

<147> 도 9 및 도 10에 도시된 것과 유사한 아키텍처가 차분 펄스 코드 변조(differential pulse code modulation)에 사용될 수 있으며, 이 경우 인코더는 시간 샘플과 예측자 간의 차이를 계산하고 예측자는 예측 인자 및 버퍼링되고, 역양자화된 시간 샘플에 기초한다. 이 예측은 일반적으로 상세한 예측자(detailed predictor)를 사용하며, 이는 설계하기가 어렵고 종종 불안정하며 막대한 시그널링 및 재구성 로직을 필요로 한다. 게다가, 이러한 방식의 압축 효율이 좋지 않다.

<148> 요약하면, 계수 예측 기법 및 도구에 의해 해소될 수 있는 몇가지 문제점에 대해 기술하였다. 그렇지만, 이러한 계수 예측 기법 및 도구가 이들 문제 모두를 해소하기 위해 적용될 필요는 없다.

<149> **B. 예시적인 계수 예측 아키텍처**

<150> 어떤 실시예들에서, 인코더는 인코딩 동안에 양자화된 스펙트럼 계수에 대해 계수 예측을 수행하고, 대응하는 디코더는 디코딩 동안에 양자화된 스펙트럼 계수에 대한 계수 예측을 수행한다. 어떤 패턴 및 유형의 컨텐츠의 경우, 계수 예측은 차후의 엔트로피 인코딩의 효율을 향상시키기 위해 스펙트럼 계수에서의 중복성(redundancy)을 감소시킨다. 예측은 가역적이며, 디코딩 동안에, (엔트로피 디코딩 이후의) 계수 예측은 인코더에서의 계수 예측과 아주 흡사하다.

<151> 도 11은 양자화된 스펙트럼 계수의 예측을 갖는 인코더를 나타낸 것이다. 예를 들어, 이 인코더는 도 2 또는 도 4에 도시된 인코더의 수정된 버전으로서, 예측자 및 차이값을 계산하기 위한 스테이지들이 추가되어 있다. 또는, 이 인코더는 도 6에 도시된 인코더의 수정된 버전으로서, 엔트로피 코딩 이전에 전처리로서 계수 예측을 갖는다.

<152> 도 11을 참조하면, 인코더는 양자화된 스펙트럼 계수(1105)와 예측자 간의 차이(예측 잔차(prediction residual)라고도 함)를 계산한다. 예측자의 경우, (지연(1110)에서의) 시간-지연된 양자화된 스펙트럼 계수는 예측 인자(1111)와 곱해진다. 예측 인자(1111)는 부수 정보로서 비트스트림(1195)으로 신호된다. 엔트로피 인코더(1190)는 차이값을 엔트로피 인코딩하여 비트스트림(1195)으로 출력한다. 양자화된 스펙트럼 계수(1105)도 차후의 양자화된 스펙트럼 계수(1105)에 대한 예측자를 계산하기 위해 제1 지연(1110)에 버퍼링된다.

<153> 도 12는 양자화된 스펙트럼 계수의 예측을 갖는 대응하는 디코더를 나타낸 것이다. 예를 들어, 디코더는 도 3 또는 도 5에 도시된 디코더의 수정된 버전으로서, 예측자를 계산하고 이 예측자를 차이값과 합성하는 스테이지들이 추가되어 있다. 또는, 이 디코더는 도 7에 도시된 디코더의 수정된 버전으로서, 엔트로피 디코딩 이후의 전처리로서 계수 예측을 갖는다.

<154> 도 12를 참조하면, 엔트로피 디코더(1290)는 비트스트림(1295)으로부터의 차이값을 디코딩한다. 이 디코더는 예측자를 계산하고 차이값을 예측자와 합성하여, 양자화된 스펙트럼 계수(1205)를 생성한다. 예측자의 경우, (지연(1210)에서의) 시간-지연된 양자화된 스펙트럼 계수가 예측 인자(1211)와 곱해진다. 예측 인자(1211)가 비트스트림(1295)으로부터 파싱된다. 양자화된 스펙트럼 계수(1205)도 차후의 양자화된 스펙트럼 계수(1205)에 대한 예측자의 계산을 위해 제1 지연(1210)에 버퍼링된다.

<155> 도 11 및 도 12에서, 인코더에서의 예측 및 차(difference) 연산 및 디코더에서의 예측 및 합 연산이 양자화된 영역에서 행해진다. 이들 연산이 동일한 영역에서 행해지는 한, 이것은 인코더 및 디코더 설계 및 복잡도를 단순화시킨다.

<156> 어떤 구현들에서, 예측, 합 및 차 연산은 정수값에 대해 행해진다. 이것은 일반적으로 구현을 단순화시키는데, 그 이유는, 부동 소수점 연산과 달리, 이들 연산이 정수 연산으로 수행될 수 있기 때문이다. 예측을 더욱 단순화시키기 위해, -1 내지 1 범위에 있는 예측 인자가 0.25의 균일한 스텝 크기를 사용하여 양자화될 수 있다. 이어서, 예측자에 대한 곱셈 연산은 이진 시프트/가산 연산을 사용하여 구현될 수 있다.

<157> 도 11 및 도 12에서, 예측자는 1차 예측자로서, 다시 말하지만 인코더/디코더("코덱") 시스템의 복잡도를 감소시킨다. 적응적 1차 예측자인 경우, 예측 인자가 변하며, 따라서 동일한 예측 인자가 장기간 동안 사용될 필요가 없다. 1차 예측자에 대해, 안정성 테스트는 별로 중요하지 않다. 예를 들어, 인코더는 단지 예측 인자가

-1 내지 +1 범위(경계 포함) 내에 있도록 제약할 뿐이다. 다른 대안으로서, 예측자는 고차 예측자이다. 예를 들어, 예측자는 16차 예측자의 경우 16개까지의 예측 인자를 갖는다.

<158> 적응적 계수 예측에 있어서, 인코더는 서브-프레임마다 또는 어떤 다른 방식으로 예측 인자를 변경한다. 예를 들어, 인코더는 서브-프레임을 다수의 균일한 크기의 세그먼트로 분할하고, 세그먼트마다 예측 인자를 계산한다. 시그널링에 관해서는, 인코더는 서브-프레임에 대한 세그먼트의 수는 물론 예측 인자도 신호한다. 따라서, 2048개 스펙트럼 계수의 서브-프레임이 16개 세그먼트로 분할되는 경우, 인코더는 세그먼트의 수 및 128개 계수의 세그먼트마다의 예측 인자를 신호한다. 서브-프레임마다의 세그먼트의 수는 시퀀스마다 한번씩, 서브-프레임마다 한번씩, 또는 어떤 다른 방식으로 신호된다. 다른 대안으로서, 세그먼트는 가변 길이를 가지며 및/또는 인코더는 다른 메카니즘을 사용하여 예측 인자를 신호한다(예를 들어, 예측 인자의 변화만을 신호하거나, 예측 인자 및 이 예측 인자가 사용되는 세그먼트의 수를 신호한다).

<159> 어떤 입력에 대해서는, 계수 예측이 성능을 향상시키지 않는다. 세그먼트별로 계수 예측을 디스에이블(disable)시키는 것은 별도로 하고(이하에서 기술됨), 인코더 및 디코더는 (예를 들어, 시퀀스 계층 온/오프 플래그(sequence layer on/off flag)로) 전체 시퀀스에 대해 또는 어떤 다른 레벨에서 계수 예측을 디스에이블시킬 수 있다.

<160> 다중-채널 오디오에 대해 계수 예측이 사용될 때, 인코딩 동안에 양자화 등이 다중-채널 변환 이후에 있는 경우, 계수 예측이 코딩된 채널마다 행해진다. 디코딩 동안에도, 계수 예측이 코딩된 채널마다 행해진다. 따라서, 이러한 다중-채널 오디오의 경우, 세그먼트마다 또는 서브-프레임마다 신호되는 예측 정보는 일반적으로 특정의 코딩된 채널의 세그먼트마다 또는 서브-프레임마다 신호된다. 계수 예측은, 시퀀스 레벨에서 또는 어떤 다른 레벨에서, 코딩된 채널마다 선택적으로 디스에이블될 수 있다. 계수 예측이 다중-채널 오디오에 대해 사용될 때, 서브-프레임마다의 세그먼트의 수가 코딩된 채널마다, 코딩된 채널의 서브-프레임마다, 또는 어떤 다른 레벨에서 신호될 수 있다.

<161> 어떤 경우에, 계수 예측은 주로 하위 및 중간 주파수에서의 스펙트럼 계수에 대한 인코딩 이득을 제공한다. 따라서, 상위 주파수에서의 스펙트럼 계수에 대해 계수 예측이 자동적으로 디스에이블될 수 있다. 또는, 계수 예측으로부터의 인코딩 이득이 주로 특정의 주파수 서브-영역에서의 스펙트럼 계수에 대한 것인 경우, 계수 예측이 이들 주파수 서브-영역에서는 선택적으로 인에이블될 수 있고 다른 곳에서는 디스에이블될 수 있다.

<162> **C. 인코딩 동안의 예시적인 계수 예측 기법**

<163> 도 13은 인코딩 동안에 양자화된 스펙트럼 계수를 예측하는 기법(1300)을 나타낸 것이다. 예를 들어, 도 1에 도시된 것 등의 인코더는 기법(1300)을 수행한다. 다른 대안으로서, 다른 인코더가 기법(1300)을 수행한다.

<164> 맨 먼저, 인코더는 오디오의 세그먼트에 대한 예측 인자를 계산한다(1310). 일반적으로, 인코더는 몇가지 기법들 중 임의의 기법을 사용하여 예측 인자를 계산한다. 예를 들어, 1차 예측자의 경우, 인코더는 가능한 예측 인자들에 대한 전수적인 검색을 수행하여 최종적인 예측 인자(예를 들어, 가장 적은 엔트로피 코딩된 비트가 얻어지는 예측 인자)를 찾아낸다. 또는, 인코더는 예측 인자를 도출하기 위해 세그먼트의 양자화된 스펙트럼 계수에 대한 상관 상수(correlation constant)(즉, E{x[i-1]x[i]}/E{x[i]x[i]})를 계산한다. 또는, 고차 예측자의 경우, 인코더는 선형 예측 계수 알고리즘(예를 들어, 자기 상관(autocorrelation) 및 자기 공분산(autocovariance)의 계산을 수반함)을 사용하며, 안정성이 요구되지 않는다. 또는, 필터의 차수 및 정밀도가 유동적인 경우, 인코더는 세그먼트에 대한 예측자 차수(1차, 2차, 3차, 기타) 및 예측 인자값과 정밀도를 계산한다. 다른 대안으로서, 인코더는 어떤 다른 메카니즘을 사용하여 예측 인자를 계산한다.

<165> 많은 경우에, 양자화된 스펙트럼 계수는 서브-프레임의 전체 스펙트럼에 걸쳐 균일한 상관을 나타내지 않는다. 이러한 상황에서 예측을 향상시키기 위해, 인코더는 스펙트럼 세그먼트별로 예측 인자를 변경할 수 있다. 예를 들어, 인코더는 서브-프레임에 대한 전체 스펙트럼을 다수의 균일한 크기의 세그먼트로 분할하고 세그먼트마다 예측 인자를 계산한다. 다른 대안으로서, 인코더는 서브-프레임의 전체 스펙트럼이거나 다른 스펙트럼 계수 블록인 세그먼트에 대한 예측 인자를 계산하거나 스펙트럼을 어떤 다른 방식으로 분할한다.

<166> 인코더는 세그먼트에 대한 예측 인자 정보를 신호한다(1320). 예를 들어, 인코더는 예측 인자를 양자화하고 이를 비트스트림으로 신호한다. 예측 인자는 엔트로피 코딩될 수 있다. 인코더는 세그먼트별로 디코딩할 시에 계수 예측을 선택적으로 디스에이블하기 위해 예측 인자 정보의 일부로서 온/오프 비트를 신호할 수 있다. 표 2는 -1 내지 1 범위에 있는 예측 인자가 0.25의 균일한 스텝 크기를 사용하여 양자화되는 구현에서 예측 인자에 대한 비트 표현을 나타낸 것이다.

<167> <표 2> 예측 인자(부수 정보)의 표현

<168>

예측 인자	이진 표현
-1.00	1000
-0.75	1001
-0.50	1010
-0.25	1011
0.00	0
0.25	1100
0.50	1101
0.75	1110
1.00	1111

<169> 다른 대안으로서, 예측 인자 정보는 어떤 다른 표현을 사용하여 신호된다.

<170> 상기한 바와 같이, 모든 세그먼트가 스펙트럼 계수 예측으로부터 이득을 볼 수 있는 것은 아니다. 0의 예측 인자는 세그먼트에 대한 예측을 효과적으로 디스에이블시키며, 예측자는 가중치를 부여받지 않고 계산될 필요가 없다. 표 2에 나타낸 코드에 있어서, 0의 예측 인자를 신호하는 데 사용되는 단일 비트 심볼이 영향을 받는 세그먼트에 대한 온/오프 비트로서 기능한다. 0 예측자가 가장 통상적인 예측 인자일 때, 단일 비트를 갖는 0 예측자를 신호하는 것은 온 비트를 절감시킨다.

<171> 상기한 바와 같이, 고차 예측자가 허용된다. 예를 들어, 고차 예측자에 대한 예측 인자 정보를 신호하기 위해, 인코더는 먼저 예측자의 차수 및 정밀도를 전송하고, 이어서 예측 인자를 하나씩 전송한다.

<172> 이어서, 인코더는 스펙트럼 계수 예측이 세그먼트에 대해 사용되는지 여부를 판정한다(1330). '예'인 경우, 인코더는 세그먼트에서 하나 이상의 양자화된 스펙트럼 계수를 예측하고(1340), 이어서 예측 코딩된(predictively coded) 계수(들)을 엔트로피 코딩한다(1350). 예를 들어, 인코더는 계수 예측을 위해 도 11에 도시된 바와 같이 지연 버퍼 및 산술을 사용한다. 다른 대안으로서, 인코더는 어떤 다른 예측 메카니즘을 사용한다. (예측(1340) 및 차후의 엔트로피 코딩(1350)은 어떤 유형의 엔트로피 코딩(1350)에 대해 반복적으로 진행될 수 있지만, 보다 일반적으로 벡터 가변 길이 코딩, 런-레벨 코딩, 또는 어떤 다른 유형의 엔트로피 코딩에 대해 일괄처리(batch)된다.)

<173> 인코더가 계수 예측(1340)을 건너뛰는 경우, 인코더는 단지 하나 이상의 양자화된 스펙트럼 계수를 엔트로피 코딩한다(1350). 다른 대안으로서, 예측 인자가 0일 때, 인코더는 예측 코딩 경로를 따라간다.

<174> 인코더는 이어서 그 다음 세그먼트에 대해 계속할지 이 기법(1300)을 종료할지를 판정한다(1360). 인코더가 계속하는 경우, 인코더는 그 다음 세그먼트에 대한 예측 인자를 계산하고(1310), 예측 인자 정보를 신호하며(1320), 이하 마찬가지로 한다.

<175> 도 13은 세그먼트별로 예측 인자를 계산하고 신호하는 것을 나타낸 것이며, 여기서 세그먼트의 수는 미리 정해져 있으며 신호되지 않는다. 다른 대안으로서, 예측 인자가 계산되고 신호되는 세그먼트의 수가 유동적이다. 이것은 일반적으로 세그먼트 정보를 규정하는 데 있어서의 증가된 비트 오버헤드의 대가로 예측 정확도를 향상시킨다. 서브-프레임 또는 다른 블록에 대해, 인코더는 균일 또는 비균일 세그먼트화를 찾아내고(예를 들어, 이 결과, 최소 비트수가 얻어짐), 세그먼트의 총수 및/또는 다른 세그먼트화 정보가 비트 스트림으로 신호된다.

<176> **D. 디코딩 동안의 예시적인 계수 예측 기법**

<177> 도 14는 디코딩 동안에 양자화된 스펙트럼 계수를 예측하는 기법(1400)을 나타낸 것이다. 예를 들어, 도 12에 도시된 것과 같은 디코더가 기법(1400)을 수행한다. 다른 대안으로서, 다른 디코더가 기법(1400)을 수행한다.

<178> 맨 먼저, 디코더는 오디오의 세그먼트에 대한 예측 인자 정보를 가져온다(1410). 예를 들어, 디코더는 비트스트림으로부터 예측 인자 정보를 파싱하고 예측 인자를 재구성한다. 예측 인자가 엔트로피 코딩되어 있는 경우, 디코더는 예측 인자를 엔트로피 디코딩한다. 디코딩 동안에 계수 예측을 선택적으로 인에이블/디스에이블하기 위해, 인코더가 예측 인자 정보의 일부로서 온/오프 비트를 신호하는 경우, 디코더는 온/오프 비트를 받는다. 따라서, 디코더는 스펙트럼 세그먼트별로 예측 인자를 변경할 수 있으며, 여기서 세그먼트는 구현에 따라 서브-프레임 또는 다른 블록의 전체 스펙트럼의 전부 또는 일부이며, 예측 인자 정보는 도 13을 참조하여 상기한 메

카니즘들 중 임의의 메카니즘을 사용하여 신호된다.

<179> 디코더는 세그먼트의 하나 이상의 양자화된 스펙트럼 계수에 대한 정보를 엔트로피 디코딩한다(1420). 인코딩 동안에 계수 예측이 사용되었을 때, 그 정보는 양자화된 스펙트럼 계수(들)에 대한 예측 잔차(들)(차이값(들))이다. 인코딩 동안에 계수 예측이 사용되지 않았을 때(0 예측자), 그 정보는 양자화된 스펙트럼 계수 자체이다.

<180> 이어서, 디코더는 세그먼트에 대해 스펙트럼 계수 예측이 사용되었는지 여부를 판정한다(1430). '예'인 경우, 디코더는 세그먼트에서의 양자화된 스펙트럼 계수(들)를 예측한다(1440). 예를 들어, 디코더는 계수 예측을 위해 도 12에 도시된 바와 같은 지연 버퍼 및 산술을 사용한다. 다른 대안으로서, 디코더는 어떤 다른 예측 메카니즘을 사용한다. (엔트로피 디코딩(1420) 및 예측(1440)은 어떤 유형의 엔트로피 디코딩(1420)을 위해 반복적으로 계속되지만, 보다 일반적으로는 벡터 가변 길이 디코딩, 런-레벨 디코딩, 또는 어떤 다른 유형의 엔트로피 디코딩에 대해 일괄 처리된다.)

<181> 어떤 경우에, 디코더는 디코딩 동안에 계수 예측을 건너뛰고, 단순히 양자화된 스펙트럼 계수(들)를 엔트로피 디코딩한다(1420). 다른 대안으로서, 예측 인자가 0일 때, 디코더는 예측 디코딩 경로(predictive decoding path)를 따라간다.

<182> 이어서, 디코더는 그 다음 세그먼트에 대해 계속할지 기법(1400)을 종료할지를 판정한다(1450). 디코더가 계속하는 경우, 디코더는 그 다음 세그먼트에 대한 예측 인자 정보를 얻고(1410), 이하 마찬가지로 한다.

<183> 도 14에서, 세그먼트의 수는 미리 결정되어 있으며 신호되지 않는다. 다른 대안으로서, 세그먼트의 수 및 예측 인자는 유동적이며, 디코더는 인코더에 의해 신호된 세그먼트화 정보를 파싱한다.

<184> **E. 결과**

<185> 일반적으로, 양자화된 스펙트럼 계수의 예측은 어떤 유형 및 패턴의 컨텐츠에 대한 차후의 엔트로피 인코딩의 효율을 향상시킨다. 예를 들어, 예측은 인접한 계수들 간의 중복성을 감소시켜, 차후의 벡터 가변 길이 코딩 및/또는 런-레벨 코딩을 더 효율적으로 만들어준다. 이와 반대로, MPEG TNS의 목적은 왜곡의 시간 분포를 제어하는 것이다.

<186> 양자화된 스펙트럼 계수의 예측으로 인한 코딩 효율의 개선을 측정하기 위해, 계수 예측을 사용하여 대량의 테스트 노래 모음이 인코딩되었다. 일반적인 입력 노래에 있어서, 노래에서의 대부분의 서브-프레임은 양자화된 영역에서 계수 예측을 사용하여 어떤 이득도 얻지 못했지만, 어떤 서브-프레임은 아주 엄청난 이득을 보았다. 예를 들어, 어떤 서브-프레임에 대해 생성된 비트는 양자화된 스펙트럼 계수의 예측으로 30%만큼이나 감소되었다. 어떤 노래의 경우, 계수 예측에 의한 전체적인 비트 레이트 감소가 32Kb/s의 공칭 비트 레이트로 동작하는 동안 3%이었고, 전체적인 비트 레이트 감소가 128Kb/s에서는 3.75%이었다. 전체 노래 모음에 관해서는, 전체적인 비트 레이트 감소가 대략 0.5%이었다.

<187> 많은 유형의 예측이 코딩 이득(coding gain)을 달성하기 위해 고차 예측자 또는 고정밀도를 사용하는 반면, 비교적 낮은 정밀도(예를 들어, 양자화된 예측 인자 값마다 3 비트)를 갖는 1차 예측자는 대부분의 시나리오에서 양자화된 스펙트럼 계수에 대해 성과가 아주 좋다. 양자화된 스펙트럼 계수는 보통 아주 작은 정수이며, 따라서 예측 인자 정밀도를 증가시키는 것이 반드시 예측된 값을 변경하거나 더 나은 것으로 만드는 것은 아니다. 잔차값이 엔트로피 코딩에 대해 정수이고 정수인 예측된 값을 계산하는 것이 타당하다. 게다가, 스펙트럼 계수에 고차 상관(higher order correlation)이 있을 때에도, 이 고차 상관은 일반적으로 양자화에 의해 왜곡되며, 그에 따라 고차 예측자가 필요없게 된다.

<188> 그렇지만, 어떤 인코딩 시나리오에서, 양자화 스텝 크기가 작고 양자화된 스펙트럼 계수가 큰 진폭을 가질 때, 고차 예측자 및/또는 높은 정밀도 예측 인자는 인코딩 효율의 더 많은 향상을 가져올 수 있다. 상기한 계수 예측 기법 및 도구는 일반적인 형태의 고차 예측자 및 고정밀도 예측 인자를 지원한다.

<189> **IV. 스펙트럼 계수의 인터리빙(interleaving) 또는 재정렬(reordering)**

<190> 상기한 바와 같이, 오디오 인코더는 종종 변환 코딩 및 그에 뒤이어서 양자화 및 엔트로피 코딩을 사용하여 압축을 달성한다. 어떤 패턴의 오디오 신호의 경우, 주파수 변환 이후에 스펙트럼 계수에 주기적인 패턴이 남아 있다. 이러한 중복성을 이용하여 코딩 효율을 개선하는 다양한 기법 및 도구가 기술된다. 상세하게는, 어떤 실시예들에서, 도 2, 도 4 또는 도 6에 도시된 것과 같은 인코더는 양자화된 스펙트럼 계수의 인터리빙 또는 재정렬을 수행한다. (도 3, 도 5 또는 도 7에 도시된 것과 같은) 대응하는 디코더는 양자화된 스펙트럼 계수의

인터리빙 또는 재정렬을 거꾸로 한다.

<191> **A. 예시적인 문제 영역**

<192> 종래에, 서브-프레임 또는 기타 윈도우의 스펙트럼 계수는 그들 간에 어떤 선형 상관도 갖지 않는 것으로 가정된다. 오히려, 스펙트럼 계수가 보통 어떤 고차의 통계적 관계를 갖는 것으로 가정되며, 인코더는 엔트로피 코딩 동안에 이를 이용하려고 한다.

<193> 이들 가정은 어떤 상황에서는 적용되지 않는다. 어떤 유형 및 패턴의 오디오 신호에 있어서, 서브-프레임 또는 다른 윈도우에 대한 스펙트럼 계수가 반드시 상관되어 있지 않아야 하는 것은 아니다. 이것은, 예를 들어, 오디오 신호가 시간 영역에서 주기적일 때 일어나며, 주기적인 신호의 스펙트럼 계수도 역시 주기성을 나타낸다. 실제로, 사인파 신호는 종종 이 거동을 보여주며, 어떤 비정상 상태의 신호도 마찬가지이다.

<194> 이를 설명하기 위해, 도 15a는 시간 영역에서의 주기적인 오디오 신호를 나타낸 것으로서, 샘플들의 시계열에 대한 진폭을 그래프로 나타낸 것이다. 도 15b는 DCT 연산으로부터의 대응하는 양자화된 스펙트럼 계수를 나타낸 것이다. 도 15b에서, 대략 57개 스펙트럼 계수마다 강한, 영이 아닌 피크 스펙트럼 계수가 있으며, 다른 곳에 있는 스펙트럼 계수는 대체로 0 또는 작은 값을 갖는다. 런-레벨 코딩 또는 벡터 가변 길이 코딩 등의 기법을 사용하여 이러한 종류의 주기적 패턴을 갖는 스펙트럼 계수를 직접 엔트로피 코딩하는 것은 효율적이지 않다. 상세하게는, 0 값을 갖는 피크 계수 또는 그 근방의 작은 값의 계수를 인코딩하는 것은 일반적으로 런-레벨 코딩 및 벡터 가변 길이 코딩 둘다에서 많은 비트를 사용한다. 그렇지만, 이러한 유형의 피크 패턴은 주기적인 신호에 대해 통상적인 것이다.

<195> 요약하면, 계수 재정렬 기법 및 도구에 의해 해결될 수 있는 몇가지 문제점들에 대해 기술하였다. 그렇지만, 이들 문제 모두를 해결하기 위해 이러한 계수 재정렬 기법 및 도구가 적용될 필요는 없다.

<196> **B. 스펙트럼 계수를 재정렬하는 예시적인 아키텍처**

<197> 어떤 실시예들에서, 인코더는 엔트로피 코딩 이전에 양자화된 스펙트럼 계수에 대한 재정렬(reordering)을 수행하고, 대응하는 디코더는 엔트로피 디코딩 이후에 양자화된 스펙트럼 계수에 대한 재정렬을 수행한다. 톤 또는 고조파를 갖는 주기적인 신호 등의 어떤 패턴 및 유형의 컨텐츠에 있어서, 재정렬은 차후의 엔트로피 인코딩의 효율을 향상시키기 위해 스펙트럼 계수에서의 중복성을 감소시킨다. 디코딩 동안에, (엔트로피 디코딩 이후의) 재정렬은 인코더에서의 재정렬을 보상한다.

<198> 도 16은 양자화된 스펙트럼 계수의 재정렬을 갖는 인코더를 나타낸 것이다. 예를 들어, 인코더는 도 2 또는 도 4에 도시된 인코더의 수정된 버전으로서, 스펙트럼 계수를 재정렬하는 스테이지들이 추가되어 있다. 또는, 인코더는 도 6에 도시된 인코더의 수정된 버전으로서, 엔트로피 코딩 이전에 전처리로서 재정렬을 갖는다.

<199> 도 16을 참조하면, 인코더는 양자화기로부터 양자화된 스펙트럼 계수(1605)를 수신한다. 양자화된 스펙트럼 계수는 재정렬/인터리빙 모듈(1680)에 의해 처리되고, 이 모듈(1680)은 선택에 따라서는 스펙트럼 계수(1605)의 일부 또는 그 전부를 재정렬하고 재정렬 정보를 비트스트림(1695)으로 신호한다.

<200> 양자화된 스펙트럼 계수(1605)가 엔트로피 코딩 효율을 향상시키기 위해 이용될 수 있는 주기적인 패턴을 나타내는 것으로 가정하자. 엔트로피 코딩 이전에, 양자화된 스펙트럼 계수는 계수들에서의 주기성을 고려하여 인터리빙 또는 재정렬된다. 예를 들어, 재정렬은 높은 값의 피크 계수들을 함께 클러스터링하고, 이는 그 계수들에 대한 차후의 벡터 가변 길이 코딩의 효율을 향상시키며, 재정렬은 다른 계수들(예를 들어, 피크들 사이의 0 값의 계수 및 낮은 값의 계수)을 함께 클러스터링하고, 이는 그 계수들에 대한 차후의 런-레벨 코딩의 효율을 향상시킨다.

<201> 스펙트럼 계수를 인터리빙하기 위해, 인코더는 주기적 패턴을 보여주는 세그먼트를 따라 스펙트럼 계수들을 인터리빙한다. 간단한 예로서, 인코더는, 다중-패스(multi-pass) 방식으로, 주기 내의 계수들에 걸쳐 브라우징하여, 먼저 각각의 주기에서의 첫번째 계수들을 선택하고 이어서 각각의 주기에서의 두번째 계수들을 선택하며, 이어서 각각의 주기에서의 세번째 계수를 선택하고, 이하 마찬가지로 한다. 인코더는 모든 계수가 선택될 때까지 재정렬을 계속한다. 스펙트럼 계수 계열이 4개의 주기 A, B, C 및 D를 포함하고 각각의 주기가 4개의 스펙트럼 계수를 포함하는 것으로 가정하자. 인터리빙 이전에, 그 계열은 다음과 같다.

<202> $A_0\ A_1\ A_2\ A_3\ B_0\ B_1\ B_2\ B_3\ C_0\ C_1\ C_2\ C_3\ D_0\ D_1\ D_2\ D_3$

<203> 인터리빙 이후에, 그 계열은 다음과 같다.

<204> $A_0\ B_0\ C_0\ D_0\ A_1\ B_1\ C_1\ D_1\ A_2\ B_2\ C_2\ D_2\ A_3\ B_3\ C_3\ D_3$

<205> 따라서, 재정렬된 계열은 계수 0, 4, 8 및 12를 먼저 두고, 이어서 계수 1, 5, 9, 13을 두며, 이하 마찬가지이다. 각각의 주기에서, 첫번째 계수만이 유효 값을 갖는 경우, 인터리빙 이후에, 이 계열에서 처음 4개의 계수만이 유효 값을 가지고, 다른 계수들 전부는 작은 값 또는 0의 값을 갖는다. 벡터 가변 길이 코딩은 처음 4개의 계수를 효율적으로 압축하고, 런-레벨 코딩은 나머지를 효율적으로 처리한다.

<206> 도 16으로 돌아가서, 선택적인 재정렬(1680) 이후에, 엔트로피 인코더(1690)는 (아마도 재정렬된) 스펙트럼 계수를 엔트로피 코딩한다. 인코더는 엔트로피 코딩된 정보를 비트스트림(1695)으로 신호한다.

<207> 도 17은 양자화된 스펙트럼 계수의 재정렬을 갖는 대응하는 디코더를 나타낸 것이다. 예를 들어, 디코더는 도 3 또는 도 5에 도시된 디코더의 수정된 버전으로서, 재정렬을 위한 스테이지가 추가되어 있다. 또는, 디코더는 도 7에 도시된 디코더의 수정된 버전으로서, 엔트로피 코딩 이후에 후처리로서 재정렬을 갖는다.

<208> 도 17을 참조하면, 엔트로피 디코더(1790)는 비트스트림(1795)으로부터 양자화된 스펙트럼 계수에 대한 정보를 디코딩한다. 비트스트림(1795)으로부터 파싱된 재정렬 정보를 사용하여, 재정렬/인터리빙 모듈(1780)은 선택에 따라서는 디코딩된 스펙트럼 계수의 일부 또는 그 전부를 재정렬하여, 원래의 순서로 된 양자화된 스펙트럼 계수(1705)를 생성한다. 본질적으로, 디코더에서의 재정렬은 인코더에서 수행된 재정렬을 거꾸로 한다.

<209> 상기한 예시적인 계열에서는, 주기 길이에 기초한 간단한 재정렬이 수행된다. 그렇지만, 어떤 경우에, 이러한 간단한 재정렬은 세그먼트에서의 선두 비주기 정보, 특정의 주기에서의 선두 0 또는 다른 오프셋, 및/또는 주기의 시작에서의 피크 계수들의 클러스터링을 해명하지 못한다. (이하에 기술되는) 부가적인 재정렬 정보가 이들 현상을 해결할 수 있다. 간단한 숫자 예를 제공하기 위해, 세그먼트가 128개 스펙트럼 계수를 가지며 그 계수들 중 일부에 대한 주기적인 패턴을 포함하는 것으로 가정한다. 이 주기 패턴은 10개 계수의 평균 주기 길이를 가지며, 19번째 계수에서 시작하고 102번째 계수에서 끝난다. 주기 길이의 배수의 관점에서, 개략적인 추정치로서, 첫번째 재정렬된 주기는 세그먼트의 세번째 주기(계수 20-29)이고, 마지막 재정렬된 주기는 10번째 주기(계수 90-99)이다. 세번째 주기에 대한 오프셋은 -1(20번째 계수보다는 19번째 계수에서의 주기에 대한 시작 위치를 나타냄)이고 10번째 주기에 대한 오프셋은 2이다. 적절한 경우, 다른 주기들에 대한 오프셋도 신호될 수 있다. 재정렬될 주기들이 일반적으로 다수의 피크 계수들로 시작하는 경우, 재정렬 이후에도 인접해 있어야만 하는 주기마다의 처음의 계수들의 수를 나타내는 값이 신호될 수 있다.

<210> 적응적 계수 재정렬에 있어서, 인코더는 서브-프레임마다 또는 어떤 다른 방식으로 재정렬을 변경한다. 예를 들어, 인코더는 서브-프레임을 다수의 세그먼트로 분할하고 세그먼트들 중 하나 이상에 대한 재정렬 정보를 계산하여, 세그먼트화 정보는 물론 재정렬 정보도 신호한다. 다른 대안으로서, 인코더는 다른 세그먼트화 및/또는 시그널링 메카니즘을 사용한다.

<211> 어떤 입력의 경우, 계수 재정렬이 성능을 향상시키지 않는다. 세그먼트마다 계수 재정렬을 디스에이블하는 것은 별도로 하고(이하에서 기술함), 인코더 및 디코더는 (예를 들어, 시퀀스 계층 온/오프 플래그로) 전체 시퀀스에 대해 또는 어떤 다른 레벨로 계수 재정렬을 디스에이블할 수 있다.

<212> 다중-채널 오디오에 대해 계수 재정렬이 사용될 때, 인코딩 동안에 양자화, 기타 등등이 다중-채널 변환 이후에 있는 경우에, 계수 재정렬은 코딩된 채널마다 행해진다. 디코딩 동안에, 계수 재정렬도 코딩된 채널마다 행해진다. 따라서, 이러한 다중-채널 오디오의 경우, 세그먼트마다, 서브-프레임마다, 또는 주기마다 신호되는 재정렬 정보는 일반적으로 특정의 코딩된 채널에 대해 세그먼트마다, 서브-프레임마다, 주기마다 신호된다. 다중-채널 오디오에 대해 계수 재정렬이 사용될 때, 세그먼트화 정보 및 재정렬 온/오프 정보는 코딩된 채널마다, 코딩된 채널의 서브-프레임마다, 또는 어떤 다른 레벨로 신호될 수 있다.

<213> 많은 경우에, 계수 재정렬은 주로 하위 및 중간 주파수에서의 스펙트럼 계수에 대해 인코딩 이득을 제공한다. 따라서, 상위 주파수에서의 스펙트럼 계수에 대해 계수 재정렬이 자동적으로 디스에이블될 수 있다. 또는, 계수 재정렬로부터의 인코딩 이득이 주로 특정 주파수 서브-영역에서의 스펙트럼 계수에 대한 것인 경우, 계수 재정렬이 선택적으로 그 주파수 서브-영역에서 인에이블될 수 있고 다른 곳에서 디스에이블될 수 있다.

<214> 섹션 III에 기술된 계수 예측이 계수 재정렬과 함께 사용될 수 있지만, 서로 다른 부류의 입력에 대해서는 계수 예측 및 계수 재정렬이 개별적으로 사용되는 것이 더 통상적이다. 이들이 함께 사용될 때, 인코딩 동안에 계수 예측은 재정렬 다음에 오며, 디코딩 동안에 계수 재정렬은 예측 다음에 오고, 계수 예측은 재정렬된 계수들의

적어도 일부(예를 들어, 피크 계수)에 대해 사용된다.

<215> **C. 인코딩 동안의 예시적인 계수 재정렬 기법**

<216> 도 18a는 인코딩 동안에 양자화된 스펙트럼 계수를 재정렬하는 기법(1800)을 나타낸 것이고, 도 18b 및 도 18c는 이 기법(1800)의 어떤 동작들을 수행하는 가능한 방법들을 상세히 나타낸 것이다. 예를 들어, 도 16에 도시한 것과 같은 인코더가 기법(1800)을 수행한다. 다른 대안으로서, 다른 인코더가 기법(1800)을 수행한다.

<217> 맨 먼저, 인코더는 세그먼트에 대한 재정렬 정보를 계산한다(1810). 예를 들어, 인코더는 도 18b에 나타낸 바와 같이 재정렬 정보를 계산한다(1810). 다른 대안으로서, 인코더는 다른 및/또는 부가적인 재정렬 정보를 계산한다.

<218> 도 18b를 참조하면, 인코더는 계수들이 재정렬될 세그먼트를 식별한다(1812). 예를 들어, 인코더는 주기적인 패턴을 갖는 스펙트럼 계수들의 세그먼트를 찾아낸다. 이를 설명하기 위해, 도 15b에서, 처음 800개 정도의 계수가 주기적인 패턴을 갖는다.

<219> 인코더는 세그먼트의 어떤 주기들을 재정렬로부터 배제시킨다. 예를 들어, 처음 1개 또는 2개의 주기가 다른 주기들과 비슷하지 않은 경우, 처음 1개 또는 2개의 주기가 재정렬 프로세스로부터 배제된다. 어떤 경우에, 세그먼트의 첫번째 부분이 선두 0 또는 비주기적 계수들을 포함한다. 그 자체로서, 인코더는 세그먼트에서 재정렬될 첫번째 주기를 추적한다. 이와 유사하게, 인코더는 세그먼트에서 재정렬될 마지막 주기도 추적한다.

<220> 그 다음에, 인코더는 세그먼트에 대한 주기의 길이를 식별한다(1814). 예를 들어, 인코더는 세그먼트에서의 피크의 수를 카운트하고 세그먼트 길이를 피크의 수로 나눈다. 또는, 인코더는 후보 주기 길이들에 대한 전수적인 검색을 수행한다. 또는, 인코더는 (파라미터 공간의 전수적인 검색과는 달리) 이진 세분 방법(binary refinement approach)을 사용하여 후보 주기 길이들을 검색한다. 또는, 인코더는 0 값/작은 값의 계수들의 런 길이를 평가한다. 또는, 인코더는 어떤 다른 메카니즘을 사용하여 세그먼트에 대한 주기 길이를 식별한다. 주기 길이가 정수 값으로 제한될 수 있거나, 주기 길이가 비정수 값(non-integer value)일 수도 있다. 정수 이하의 정밀도(sub-integer precision)를 허용하면 재정렬의 효율을 상당히 향상시킬 수 있으며, 궁극적으로 엔트로피 코딩 이득을 향상시킬 수 있다.

<221> 인코더는 또한 주기 조정(period adjustment) 및 프리롤 값(preroll value)을 포함할 수 있는 다른 재정렬 정보도 식별한다(1816). 예를 들어, 비정수 주기 길이를 허용하는 구현에서, 인코더는 다음과 같이 다른 재정렬 정보를 계산한다.

<222> 주기 i의 최초 시작 위치는 round(i*period_length)이고, 주기 i의 최초 종료 위치는 그 다음 주기의 최초 시작 위치이다. 인코더는 추적을 위해 주기의 시작 위치 및/또는 종료 위치를 저장하는 주기 위치 테이블을 유지한다. 이것은 또한 인코더가 다른 위치를 평가할 때 테이블에서의 주기들의 위치를 단지 조정할 수 있게만 해준다.

<223> 상세하게는, 인코더는 엔트로피 코딩을 향상시키기 위해 주기의 시작 위치 및/또는 종료 위치를 최초 위치로부터 하나 이상의 계수만큼 이동시킬 수 있다. 예를 들어, 주기의 최초 시작 위치 바로 이전에 큰 유효 계수가 있는 경우, 그 큰 유효 계수가 이전의 주기의 끝이 아니라 주기의 시작에 나타나도록, 인코더는 시작 위치를 몇 개의 계수만큼 좌측으로 시프트한다. 다른 대안으로서, 인코더는 어떤 다른 메카니즘을 사용하여 재정렬될 주기들의 시작 위치 및/또는 종료 위치에 대한 조정량을 결정한다.

<224> 인코더는 또한 프리롤 값을 선택한다. 프리롤은 서로에 대해 재정렬되지 않는 주기의 시작에 있는 계수들을 나타낸다. 통상적으로, 주기의 시작에 있는 피크는 단지 하나의 스펙트럼 계수가 아닐 뿐이다. 예를 들어, 주기의 시작에 큰 값을 갖는 2개 또는 3개의 계수가 있을 수 있으며, 이러한 계수들이 프리롤 계수이다. 프리롤 계수는 특별한 방식으로 인터리빙되어, 재정렬을 위해 그룹으로서 효과적으로 처리된다. 환언하면, 프리롤 계수들은 세그먼트의 주기들에 대한 재정렬 이후에도 인접해 있다. 프리롤 값은 재정렬될 주기들에 대한 프리롤 계수들의 수(예를 들어, 1, 2, 3)를 나타낸다. 또는, 세그먼트마다 프리롤을 계산하지 않고, 인코더는 재정렬될 주기마다 프리롤을 계산한다.

<225> 다른 대안으로서, 인코더는 어떤 다른 메카니즘을 사용하여 다른 재정렬 정보를 식별한다(1816).

<226> 도 18a로 돌아가면, 인코더는 세그먼트에 대한 재정렬 정보를 비트스트림으로 신호한다(1830). 예를 들어, 인코더는, 도 18b에 나타낸 바와 같이 계산된 재정렬 정보에 대해, 도 18c에 나타낸 바와 같이 재정렬 정보를 신

호한다(1830). 다른 대안으로서, 인코더는 다른 및/또는 부가적인 재정렬 정보를 신호한다.

<227> 도 18c를 참조하면, 인코더는 재정렬을 위한 온/오프 비트를 신호한다(1832). 예를 들어, 인코더는 계수 재정렬이 사용될 때의 비트 비용을 계수 재정렬이 사용되지 않을 때의 비트 비용과 비교한다. 인코더는 더 나은 성능을 제공하는 모드를 선택하고, 인코더는 세그먼트마다 단일 비트를 사용하여 어느 모드가 선택되는지를 나타낸다. 다른 대안으로서, 인코더는 어떤 다른 메카니즘을 사용하여 및/또는 전체 세그먼트가 아닌 어떤 기간 동안 온/오프 정보를 신호한다.

<228> 재정렬이 사용될 때(결정(1834)으로부터의 "예" 분기), 디코더는 주기 길이를 신호한다(1836). 비정수 주기 길이가 허용될 때, 주기 길이는 정수 부분 및 소수 부분으로 표현될 수 있으며, 둘다 비트스트림으로 신호된다. 정수 주기 길이(또는 비정수 주기 길이의 정수 부분)은 $\log_2$(가장 큰 주기 길이) 비트를 사용하여 고정 길이 코드(fixed length code, FLC)로서 신호된다. 예를 들어, 가장 큰 주기 길이는 128이고, 정수 주기 길이가 $\log_2(128) = 7$ 비트를 사용하여 신호된다. 소수 부분은 3-비트 FLC를 사용하여 신호될 수 있다. 다른 대안으로서, 주기 길이는 다른 메카니즘을 사용하여 신호된다.

<229> 인코더는 또한 계수들이 재정렬될 첫번째 주기를 신호한다(1838). 사실상, 이것은 재정렬을 위한 시작 위치를 개략적으로 나타낸다. 첫번째 재정렬된 주기는 주기 길이의 단위로 표현될 수 있다. 첫번째 재정렬된 주기는, 예를 들어, 3-비트 FLC로 신호되고, 이 경우 첫번째 재정렬된 주기는 세그먼트 내의 첫번째 주기부터 8번째 주기까지의 임의의 주기이다. 다른 대안으로서, 첫번째 재정렬된 주기는 다른 메카니즘을 사용하여 신호된다.

<230> 인코더는 또한 계수들이 재정렬될 마지막 주기를 신호한다(1840). 마지막 재정렬된 주기는 주기 길이의 단위로 표현될 수 있다. 마지막 재정렬된 주기는, 예를 들어, log2(주기의 최대 수) 비트를 사용하여 FLC로서 신호된다. 인코더는 세그먼트 내의 계수들의 수 및 주기 길이로부터 주기의 최대 수를 도출한다. 다른 대안으로서, 마지막 재정렬된 주기는 다른 메카니즘을 사용하여 신호된다.

<231> 인코더는 위치 조정을 신호한다(1842). 계수들이 재정렬될 주기들에 대해, 인코더는 최초 시작 위치 및/또는 종료 위치에 대한 오프셋을 나타내는 정보를 신호한다. 예를 들어, 주기마다 하나의 조정값이 신호되고, 이 조정값은 계수들의 수로서 신호된다. 이러한 조정값은 $\log_2$(오프셋 범위) 비트를 사용하여 FLC로서 신호될 수 있다. 따라서, 오프셋 범위가 16일 경우, -8...7 계수들의 조정 범위에 대해, 조정값은 $\log2(16) = 4$ 비트를 사용하여 신호된다. 다른 대안으로서, 이 조정값은 다른 메카니즘을 사용하여 신호된다(예를 들어, 이전의 조정값(절대항으로 되어 있지 않음)에 대한 조정을 신호하는 것 또는 모든 주기에 대해 하나의 조정을 신호하는 것).

<232> 인코더는 또한 프리롤 값을 신호한다(1844). 어떤 수의 계수들의 프리롤 값이 $\log_2$(가장 큰 프리롤 + 1) 비트를 사용하여 FLC로서 신호된다. 예를 들어, (0, 1, 2, 또는 3의 프리롤에 대해) 가장 큰 프리롤 길이가 3이고, 프리롤 값은 $\log2(4) =2$ 비트를 사용하여 신호된다. 다른 대안으로서, 프리롤 값은 다른 메카니즘을 사용하여 신호된다.

<233> 도 18a로 되돌아가서, 인코더는 계수 재정렬이 사용되는지 여부를 판정한다(1860). '아니오'인 경우, 인코더는, 벡터 가변 길이 코딩, 런-레벨 코딩, 또는 어떤 다른 엔트로피 코딩을 사용하여, 세그먼트의 양자화된 스펙트럼 계수를 단순히 엔트로피 인코딩한다(1880). 반면에, 계수 재정렬이 사용되는 경우, 인코더는 세그먼트의 계수들의 적어도 일부를 재정렬하고(1870) 벡터 가변 길이 코딩, 런-레벨 코딩, 또는 어떤 다른 엔트로피 코딩을 사용하여 (선택적으로) 재정렬된 계수들을 엔트로피 인코딩한다(1880). 예를 들어, 인코더는, 도 18b에 나타낸 바와 같이 계산되고 도 18c에 나타낸 바와 같이 신호되는 정보를 재정렬하기 위해, 다음과 같이 재정렬(1870)을 수행한다.

<234> 요약하면, 인코더는 계수들을 재정렬하고 이 계수들을 새로운 계수 버퍼로 (또는 재정렬 프로세스가 버퍼링을 위해 추가의 자원을 사용하지 않도록 엔트로피 코더로 직접) 출력한다. 인코더는 계수들이 재정렬될 주기의 시작 위치 및/또는 종료 위치를 나타내는 (상기한) 테이블을 브라우징한다. 일반적으로, 인코더는 첫번째 이러한 주기부터 마지막 이러한 주기까지 루프를 돈다.

<235> 어느 한 주기에 대해, 인코더는 재정렬 시에 아직 처리되지 않은 첫번째 계수를 찾아낸다. 이 계수가 프리롤 영역 내에 있는 경우, 인코더는 그 계수 및 하나 이상의 그 다음에 오는 프리롤 계수들을 그들의 원래의 순서로 출력한다. 그렇지 않은 경우, 인코더는 아직 처리되지 않은 첫번째 계수를 출력할 뿐이다. 이어서, 인코더는 주기 내의 모든 처리된 계수들을 처리된 것으로 표시한다. 인코더는그 다음 주기의 첫번째 미처리된 계수에 대

해 계속한다.

<236> 어떤 주기에 대해, 미처리된 계수가 없는 경우, 인코더는 단지 그 다음 주기로 계속간다.

<237> 인코더가 한 반복에서 처음부터 끝까지 모든 주기를 검사한 후에, 인코더는 첫번째 주기부터 되풀이한다. 궁극적으로, 인코더는 재정렬될 주기들 내의 계수들 전부를 처리한다. 세그먼트 내의 계수들이 재정렬되어 있지 않을 때, 인코더는 단지 그 계수들을 새로운 계수 버퍼로 복사(또는 적절한 때에 이들을 직접 엔트로피 코더로 전송)할 수 있을 뿐이다.

<238> 다른 대안으로서, 인코더는 어떤 다른 메카니즘을 사용하여 재정렬(1870)을 수행한다. 또는, 인코더는 다른 및/또는 부가적인 재정렬 정보에 따라 재정렬(1870)을 수행한다.

<239> 인코더는 이어서 그 다음 세그먼트에 대해 계속할지 또는 기법(1800)을 종료할지를 판정한다(1890). 인코더가 계속하는 경우, 인코더는 그 다음 세그먼트에 대한 재정렬 정보를 계산하고(1810), 이 재정렬 정보를 신호하며(1820), 이하 마찬가지로 한다.

<240> 도 18a 내지 도 18c가 재정렬 정보를 계산하는 동작들을, 별개의 것으로서 재정렬 정보를 신호하는 동작보다 이전에 있는 것으로 나타내고 있지만, 다른 대안으로서, 이러한 동작들은 서로 또는 다른 동작들과 인터리빙된다.

<241> **D. 디코딩 동안의 예시적인 계수 재정렬 기법**

<242> 도 19a는 디코딩 동안에 양자화된 스펙트럼 계수를 재정렬하는 기법(1900)을 나타낸 것이고, 도 19b 및 도 19c는 기법(1900)의 어떤 동작들을 수행하는 가능한 방법들을 상세히 나타낸 것이다. 예를 들어, 도 12에 나타낸 것과 같은 디코더가 기법(1900)을 수행한다. 다른 대안으로서, 다른 디코더가 기법(1900)을 수행한다.

<243> 맨 먼저, 디코더는 세그먼트에 대한 재정렬 정보를 얻는다(1910). 디코더는 일반적으로 인터리빙/재정렬에서 사용하기 위해 비트스트림으로부터 부수 정보를 읽어온다. 예를 들어, 디코더는, 도 18c에 나타낸 바와 같이 신호된 재정렬 정보에 대해, 도 19b에 나타낸 바와 같이 재정렬 정보를 얻는다(1910). 다른 대안으로서, 디코더는 다른 및/또는 부가적인 재정렬 정보를 얻는다.

<244> 도 19b를 참조하면, 디코더는 비트스트림으로부터 재정렬을 위한 온/오프 비트를 파싱한다(1912). 예를 들어, 디코더는 비트스트림으로부터 단일 비트를 읽어오며, 이 단일 비트는 계수 재정렬을 갖는 모드를 사용할지 계수 재정렬을 갖지 않는 모드를 사용할지를 나타낸다. 다른 대안으로서, 온/오프 정보는 어떤 다른 메카니즘을 사용하여 신호되고 파싱되며 및/또는 전체 세그먼트가 아닌 어떤 기간에 대한 것이다.

<245> 계수 재정렬이 사용될 때(결정(1914)로부터의 "예" 분기), 디코더는 비트스트림으로부터 주기 길이를 파싱한다(1916). 비정수 주기 길이가 허용될 때, 주기 길이는 정수 부분 및 소수 부분(둘다 비트스트림으로부터 파싱됨)으로 표현될 수 있다. 정수 주기 길이(또는 비정수 주기 길이의 정수 부분)는 $\log_2$(가장 큰 주기 길이) 비트를 사용하여 FLC로서 표현된다. 다른 대안으로서, 주기 길이는 다른 메카니즘으로 신호된다.

<246> 디코더는 또한 비트스트림으로부터 계수들이 재정렬될 첫번째 주기를 파싱하며(1918), 이는 재정렬을 위한 시작 위치를 개략적으로 알려준다. 첫번째 재정렬된 주기는 주기 길이의 단위로 표현될 수 있다. 첫번째 재정렬된 주기는, 예를 들어, 3-비트 FLC로 표현된다. 다른 대안으로서, 첫번째 재정렬된 주기는 다른 메카니즘을 사용하여 신호되고 파싱된다.

<247> 디코더는 또한 비트스트림으로부터 계수들이 재정렬될 마지막 주기를 파싱한다(1940). 마지막 재정렬된 주기는 주기 길이의 단위로 표현될 수 있다. 마지막 재정렬된 주기는, 예를 들어, $\log_2$(주기의 최대 수) 비트를 사용하여 FLC로서 신호되고, 여기서 디코더는 세그먼트 내의 계수들의 수 및 주기 길이로부터 주기의 최대 수를 도출한다. 다른 대안으로서, 마지막 재정렬된 주기는 다른 메카니즘을 사용하여 신호되고 파싱된다.

<248> 주기 길이, 첫번째 재정렬된 주기, 및 마지막 재정렬된 주기와 함께, 디코더는 추적을 위해 주기들의 시작 위치 및/또는 종료 위치를 저장하는 주기 위치 테이블을 채울 정보를 갖는다. 따라서, 디코더는 대응하는 인코더에 의해 사용된 주기 위치 테이블을 복원할 수 있다.

<249> 디코더는 비트스트림으로부터 위치 조정을 파싱한다(1922). 계수들이 재정렬될 주기들에 대해, 디코더는 최초 시작 및/또는 종료 위치에 대한 오프셋을 나타내는 정보를 파싱한다. 예를 들어, 주기마다 하나의 조정값이 파싱되고, 이 조정값은 계수들의 수로서 표현된다. 이러한 조정값은 $\log_2$(오프셋 범위) 비트를 사용하여 FLC로서 표현될 수 있다. 다른 대안으로서, 조정값은 다른 메카니즘을 사용하여 신호되고 파싱된다.

<250> 위치 조정 정보와 함께, 디코더는 주기 위치 테이블에 주기들의 시작 위치 및/또는 종료 위치를 조정하는 정보를 갖는다.

<251> 디코더는 또한 프리롤 값을 파싱한다(1924). 어떤 수의 계수들의 프리롤 값은 $\log_2$(가장 큰 프리롤 + 1) 비트를 사용하여 FLC로서 표현된다. 다른 대안으로서, 프리롤 값은 다른 메카니즘을 사용하여 신호되고 파싱된다.

<252> 도 19a로 되돌아가서, 디코더는 벡터 가변 길이 디코딩, 런-레벨 디코딩, 또는 어떤 다른 엔트로피 디코딩을 사용하여 비트스트림으로부터 계수 정보를 디코딩한다(1930). 디코딩에서 재정렬이 사용되지 않았을 때, 디코더는 세그먼트의 양자화된 스펙트럼 계수들을 그들의 원래의 순서로 디코딩한다(1930). 반면에, 인코딩에서 재정렬이 사용되었을 때, 디코더는 재정렬되어 있는 양자화된 스펙트럼 계수들을 디코딩한다(1930).

<253> 디코더는 또한 디코딩 동안에 계수 재정렬이 사용되는지 여부를 판정한다(1960). 디코딩 동안에 계수 재정렬이 사용되는 경우, 디코더는 엔트로피 디코딩되어 있는 세그먼트의 계수들 중 적어도 일부를 재정렬한다(1970). 예를 들어, 디코더는, 도 19b에 나타낸 바와 같이 검색된 정보를 재정렬하기 위해, 다음과 같이 재정렬(1970)을 수행한다.

<254> 디코더는 세그먼트에 대한 재정렬 정보(예를 들어, 주기 길이, 첫번째 재정렬된 주기, 마지막 재정렬된 주기)로부터 주기 위치 테이블을 발생하고(1972), 테이블에 주기 조정을 적용한다(1974). 이 테이블은 재정렬에서 사용하기 위한 주기들의 시작 위치 및/또는 종료 위치를 저장한다. 다른 대안으로서, 디코더는 테이블 발생 프로세스를 건너뛰거나 어떤 다른 테이블 구조를 사용한다.

<255> 디코더는 이어서 주기 위치 테이블 및 프리롤 값을 사용하여 계수들을 재정렬한다(1976). 요약하면, 디코더는 계수들을 재정렬하고 그 계수들을 새로운 계수 버퍼로 출력하여, 인코딩 동안에 수행된 재정렬을 거꾸로 한다. (다른 대안으로서, 디코더는 엔트로피 디코더의 출력을 직접 재정렬할 수 있으며, 따라서 계수 버퍼링을 위한 부가적인 자원이 전혀 사용되지 않는다.) 디코더는 계수들이 재정렬되어야만 하는 주기들의 시작 위치 및/또는 종료 위치를 나타내는 (상기한) 주기 위치 테이블을 사용한다. 일반적으로, 디코더는 엔트로피 디코딩된 스펙트럼 계수를 엔트로피 디코딩의 결과로부터 얻어지는 순서로 처리한다. 예를 들어, 첫번째 재정렬된 주기에 대한 위치에, 디코더는 첫번째 미처리된 계수는 물론 첫번째 재정렬된 주기에 대한 프리롤 영역 내의 임의의 미처리된 계수들을 둔다. 그 다음에, 두번째 재정렬된 주기에 대한 위치에, 디코더는 그 다음 미처리된 계수는 물론 두번째 재정렬된 주기에 대한 프리롤 영역 내의 임의의 미처리된 계수들을 둔다. 디코더는 마지막 재정렬 주기까지 주기들 각각에 대해 이 프리롤 처리를 반복한다. 이어서, 디코더는 반복하여 연속적인 미처리된 계수들을 첫번째, 두번째, 세번째,... 재정렬된 주기에 대한 위치에 두고, 재정렬된 주기가 채워졌을 때 그 재정렬된 주기를 건너뛴다. 궁극적으로, 디코더는 재정렬될 주기들 내의 계수들 전부를 처리한다. 세그먼트 내의 계수들이 재정렬되어 있지 않을 때, 디코더는 단지 그 계수들을 새로운 계수 버퍼 내의 대응하는 위치로 복사만 할 수 있다.

<256> 다른 대안으로서, 디코더는 어떤 다른 메카니즘을 사용하여 재정렬(1970)을 수행한다. 예를 들어, 주기 위치 테이블 및 프리롤 값을 사용하여, 디코더는 엔트로피 디코딩된 계수들 전체를 브라우징하고, 첫번째 재정렬된 주기에 대한 스펙트럼 계수들을 선택하여 출력한다. 이어서, 인코더는 엔트로피 디코딩된 계수들 전체를 브라우징하고, 두번째 재정렬된 주기에 대한 스펙트럼 계수들을 선택하여 출력하며, 마지막 재정렬된 주기까지 마찬가지로 한다. 또는, 디코더는 다른 및/또는 부가적인 재정렬 정보에 따라 재정렬(1970)을 수행한다.

<257> 디코더는 이어서 그 다음 세그먼트를 계속할지 기법(1900)을 종료할지를 판정한다(1990). 디코더가 계속하는 경우, 디코더는 그 다음 세그먼트에 대한 재정렬 정보를 얻으며(1910), 이하 마찬가지로 한다.

<258> 도 19a 내지 도 19c가 재정렬 정보를 얻는 동작을, 별개의 것으로서 재정렬의 다른 동작들보다 이전에 있는 것으로 도시하고 있지만, 다른 대안으로서, 이들 동작은 서로 또는 다른 동작들과 인터리빙된다.

<259> **E. 결과**

<260> 일반적으로, 양자화된 스펙트럼 계수의 재정렬은 주기적인 신호에 대한 차후의 엔트로피 인코딩의 효율을 향상시킨다. 예를 들어, 재정렬은 유사한 값을 갖는 계수들을 로컬적으로 그룹화하고, 차후의 벡터 가변 길이 코딩 및/또는 런-레벨 코딩을 더 효율적으로 만들어준다.

<261> 상기한 재정렬은 구현하기가 비교적 간단하고 낮은 계산 복잡도를 갖는다. 메모리 사용에 관해서는, 어떤 구현에서, 재정렬 동작에 의해 요구되는 유일한 추가의 메모리는 주기 위치 테이블이며, 이는 아주 작다.

<262> 도 20은 계수 재정렬 이후의 도 15b의 스펙트럼 계수를 나타낸 것이다. 주기 길이는 56.7이다. 재정렬은 위치 114에서 시작하고(세그먼트에서의 세번째 주기를 시작함), 재정렬은 위치 1021 근방에서 종료한다(세그먼트의 18번째 주기를 종료함). 세그먼트에서의 주기들에 대해 프리롤은 3이다. 재정렬 이후에, 약 위치 250까지의 계수들은 벡터 가변 길이 코딩에 잘 맞으며, 그 이후의 계수들은 런-레벨 코딩에 잘 맞는다.

<263> 재정렬로 인한 코딩 이득은 신호의 주기성에 의존한다. 신호가 시간 영역에서 주기적인 경우, 종종 스펙트럼 계수의 재정렬로부터 상당한 이득이 있다. 그렇지 않은 경우, 코딩 이득이 일반적으로 그다지 크지 않거나 전혀 없다. 도 21은 주기적인 신호를 갖는 한 예시적인 오디오 파일의 서브-프레임마다의 재정렬로 인한 코딩 이득을 나타낸 것이다. 서브-프레임에 대한 가장 큰 이득은 40%를 넘으며, 파일에 대한 평균 이득은 약 11%이다.

<264> **V. 적응적 코딩/디코딩에서 다수의 엔트로피 모델의 선택적 사용**

<265> 어떤 실시예들에서, 도 2, 도 4 또는 도 6에 도시한 것과 같은 인코더는 인코더가 다수의 엔트로피 모델을 선택적으로 사용하는 적응적 엔트로피 코딩을 수행한다. (도 3, 도 5 또는 도 7에 도시한 것과 같은) 대응하는 디코더는 디코더가 다수의 엔트로피 모델을 선택적으로 사용하는 적응적 엔트로피 디코딩을 수행한다. 다수의 엔트로피 모델의 선택적 사용을 위한 기법 및 도구는, 오디오, 비디오, 이미지 또는 임의의 다른 데이터의 무손실 및 손실 압축 및 압축해제를 비롯한, 심볼값이 다수의 확률 분포를 갖는 다양한 시나리오에서 적용가능하다.

<266> **A. 예시적인 문제 영역**

<267> 심볼의 적응적 코딩은 종종 심볼값의 확률 분포가 변할 때 엔트로피 코딩의 효율을 향상시키는 데 사용된다. 적응적 산술 코딩은 서로 다른 또는 변하는 확률 분포를 직접 사용할 수 있다. (적응적 허프만 코딩과 같은) 적응적 가변 길이 코딩의 경우, 심볼값에 대한 서로 다른 엔트로피 모델이 서로 다른 또는 변하는 VLC 테이블로 구현된다.

<268> 후방 적응(backward adaptation)의 경우, 코딩/디코딩이 이미 처리된 심볼에 기초하여 적응한다. 전방 적응(forward adaptation)의 경우, 그 적응을 기술하는 정보가 명시적으로 신호된다. 예를 들어, 일련의 심볼에 사용될 VLC 테이블을 알려주기 위해 테이블 전환 코드(table switch code)가 신호된다.

<269> 확률 분포(또는 가변 길이 코딩/디코딩에 대해 사용되는 대응하는 VLC)를 동적으로 변화시킴으로써 적응이 달성된다. 또는, 고정된 일련의 서로 다른 사전-훈련된 확률 분포(또는 대응하는 VLC 테이블) 중에서 선택함으로써 적응이 달성될 수 있다.

<270> 다수의 서로 다른 분포/VLC 테이블을 사용하는 것의 한가지 단점은 인코더 및 디코더에 메모리가 필요하다는 것인데, 그 이유는 사용되는 메모리가 분포/VLC 테이블의 수에 따라 선형적으로 증가하기 때문이다. 예를 들어, 16개의 VLC 테이블이 사용되는 경우, 인코더 및 디코더에서의 VLC 테이블을 위해 단일의 VLC 테이블의 경우와 비교하여 대략 16배의 메모리가 사용된다.

<271> 요약하면, 다수의 엔트로피 모델의 선택적 사용을 위한 기법 및 도구가 해결할 수 있는 문제점에 대해 기술하였다. 그렇지만, 이러한 기법 및 도구가 이 문제를 해결하기 위해 적용될 필요는 없다.

<272> **B. 다수의 엔트로피 모델의 선택적 사용**

<273> 다수의 엔트로피 모델의 선택적 사용은 다수의 분포/VLC 테이블에 대한 자원 사용을 상당히 감소시킬 수 있다. 이와 동시에, 다수의 엔트로피 모델의 사용과 연관된 인코딩 이득의 대부분이 여전히 달성될 수 있다. 다양한 통상적인 시나리오에서, 다수의 엔트로피 모델의 선택적 사용은 전부가 아닌 일부의 심볼값에 대해 서로 다른 분포/VLC 테이블 중에서 선택하는 것을 수반한다. 보다 일반적으로, 이는 어떤 심볼값에 대해서는 더 많은 적응성(adaptivity)을 가능하게 해주고 다른 심볼값에 대해서는 더 적은 적응성을 가능하게 해주도록 계층적으로 구성되어 있는 서로 다른 분포/VLC 테이블 중에서 선택하는 것을 수반한다.

<274> 어떤 테스트에 따라, 일련의 심볼값이 어떤 확률이 높은 심볼값과 어떤 확률이 낮은 심볼값을 포함하는 것으로 가정한다. 분포/테이블에 사용되는 메모리를 감소시키기 위해, 인코더 및 디코더는 확률이 높은 심볼값에 대해서는 다수의 분포/테이블을 사용하지만, 확률이 낮은 심볼값은 다수의 분포/테이블로 표현되지 않는다. 이것은 다수의 분포/테이블에 사용되는 메모리를 감소시키는데, 코딩 이득에 대한 불이익은 무시할 정도이다. (많은 상황에서, 비교적 적은 수의 심볼값들이 확률 분포의 많은 부분을 차지한다.) 상세하게는, 엔트로피 모델이 주어진 적응 상태에 대해 조건적인 것으로 생각되는 경우, 각자의 서로 다른 상태에서 확률이 높은 심볼값에 대해 다른 분포가 있다. 그렇지만, 서로 다른 상태에서 확률이 낮은 심볼값에 대한 상대 분포는 동일하다.

<275> 일련의 256개 심볼값에 대해, 심볼값 중 32개가 거의 대부분 사용되는 경우, 인코더 및 디코더는 32개 심볼값에 대한 6개의 VLC 테이블 간에 전환할 수 있으며, 이 때 6개의 VLC 테이블 각각은 또한 나머지 224개 심볼값에 대한 단일의 VLC 테이블로 전환하기 위한 이스케이프 코드(escape code)를 포함한다.

<276> 또는, 일련의 256 심볼값에 대해, 심볼값 중 7개가 거의 대부분 사용되고, 심볼값들 중 21개가 때때로 사용되며, 나머지 심볼들은 단지 드물게 사용되는 것으로 가정하자. 인코더 및 디코더는 7개의 가장 흔한 심볼값에 대해 11 VLC 테이블 간에 전환할 수 있으며, 이 경우 11개 VLC 테이블 각각은 21개의 그 다음으로 가장 흔한 심볼값에 대해 2개의 VLC 테이블 간에 전환하기 위한 이스케이프 코드를 포함한다. (이스케이프 코드에 뒤이어서 전방 적응을 위한 테이블 선택 정보가 올 수 있다.) 21개의 심볼값에 대한 2개의 VLC 테이블 각각은 나머지 심볼값들에 대한 VLC 테이블로의 전환을 위한 이스케이프 코드를 포함한다.

<277> 도 22는 엔트로피 모델/상태(예를 들어, 분포, VLC 테이블)의 계층적 구성의 관점에서 더 복잡한 예를 나타낸 것이다. 인코더 및 디코더는 심볼값 B, F, H 및 I에 대해 8개의 엔트로피 모델을 사용하며, 여기서 8개의 엔트로피 모델 각각은 또한 2개의 전환점(switch point)을 포함한다. 예를 들어, 인코더 및 디코더가 엔트로피 모델에 대한 확률 분포를 사용하는 경우, 전환점은 분포에서의 특수한 전환 확률값이다. 인코더 및 디코더가 엔트로피 모델에 대해 VLC 테이블을 사용하는 경우, 전환점은 이스케이프 코드 또는 다른 특수한 VLC이다. 8개의 엔트로피 모델에서, 첫번째 전환점은 심볼값 A 및 C에 대한 엔트로피 모델로 전환하기 위한 것이고, 두번째 전환점은 심볼값 D, E, G, J 및 K에 대한 엔트로피 모델로 전환하기 위한 것이다.

<278> 인코더 및 디코더는 심볼값 A 및 C에 대해 3개의 엔트로피 모델을 사용한다. 인코더 및 디코더는 심볼값 E, J 및 K에 대해 4개의 엔트로피 모델을 사용하며, 여기서 4개의 엔트로피 모델 각각도 전환점을 포함한다. 이 전환점은 심볼값 D 및 G에 대한 엔트로피 모델로 전환하기 위한 것이다.

<279> 도 22에서, 심볼값들의 부분집합(subset)은 그의 포함집합(superset)보다 더 적은 연관된 엔트로피 모델을 갖는다. 이것은 확률이 더 높은 심볼값에 대해 더 많은 적응성이 가능하게 되고 확률이 낮은 심볼값에 대해 더 적은 적응성이 가능하게 되는 많은 통상적인 시나리오와 일치한다. 그렇지만, 다른 대안으로서, 부분집합이 그의 포함집합보다 더 많은 연관된 엔트로피 모델을 가질 수 있다.

<280> 다수의 엔트로피 모델 중에서의 선택은 후방 적응 메카니즘(backward adaptive mechanism) 또는 전방 적응 메카니즘(forward adaptive mechanism)을 통할 수 있다. 다수의 엔트로피 모델 자체가 고정되고 사전-훈련될 수 있거나 이들이 동적으로 변할 수 있다. 엔트로피 모델은 다양한 엔트로피 코딩 및 디코딩 방식에서 적용될 수 있다. 산술 코딩 및 디코딩은 심볼값들의 전부가 아닌 일부에 대해 다수의 확률 분포를 선택적으로 사용할 수 있다. 또는, 가변 길이 코딩 및 디코딩은 심볼값들의 전부가 아닌 일부에 대해 다수의 VLC 테이블을 사용할 수 있다.

<281> **1. 상태에 대한 분포의 조정**

<282> 인코더 또는 디코더가 (심볼값들의 전부가 아닌) 일부 심볼값들에 대해 다수의 엔트로피 모델을 선택적으로 사용하기 위해서, 다수의 엔트로피 모델이 그에 따라 조정된다. 이하의 분석은, 간단한 예를 참조하여, 일련의 상태에 대한 실제 확률 분포에 대한 조정을 나타낸 것이다.

<283> $X(i) = X(0), X(1), \ldots, X(M\text{-}1)$로 표시된 M개의 심볼값의 분포를 적응시키기 위한 $S(j) = S(0), S(1), \ldots, S(N\text{-}1)$로 표시된 N개의 상태가 있는 것으로 가정한다.

<284> P_S는 상태에 대한 확률 분포를 나타내고, $P_{S(j)}$는 상태가 $S(j)$일 확률이다. $P_{S(j),X}$는 상태 $S(j)$에 있을 때 심볼값에 대한 확률 분포를 나타내고, $P_{S(j),X(i)}$는 상태 $S(j)$에 있을 때 심볼이 값 $X(i)$을 가질 확률이다. M개의 심볼값 중에서, L개의 심볼값이 확률이 높은 것으로 지정되고, M-L개의 심볼값이 확률이 낮은 것으로 지정된다. L개의 확률이 높은 심볼값의 집합이 집합 Q이고, M-L개의 확률이 낮은 심볼값의 집합이 집합 R이다.

<285> 확률이 높은 심볼값과 확률이 낮은 심볼값의 지정은 구현 의존적이며 유동적이지만, 적절한 지정은 보다 효율적인 코딩을 가져온다. 모든 상태 $S(j)$에 대해 $P_{S(j),X(q)} > P_{S(j),X(r)}$일 필요는 없으며, 여기서 $X(q)$는 Q

내의 심볼값을 나타내고, $X(r)$은 R 내의 심볼값을 나타낸다. 환언하면, 모든 상태에서 주어진 "확률이 높은" 심볼값이 주어진 "확률이 낮은" 심볼값보다 더 높은 확률을 가질 필요는 없다.

<286> 상태 $S(j)$에 대한 수정된 분포 $P'_{S(j),X}$는 상태 $S(j)$에 대한 실제 심볼값 분포 $P_{S(j),X}$와 비슷하다. (1) 집합 R 내의 심볼값 $X(i)$에 대한 조건부 분포 $P'_{S(j),X(i),R}$가 모든 $S(j)$에 대해 동일하지만 (2) 집합 Q 내의 심볼값에 대한 분포가 임의의 주어진 $S(j)$에 대해 변하지 않도록 하는 $P'_{S(j),X}$는 $P_{S(j),X}$와 비슷하다(집합 Q 내의 심볼값 $X(i)$에 대해 $P'_{S(j),X(i)} = P_{S(j),X(i)}$이다).

<287> N=3이고 M=5인 것으로 가정하자. 상태들의 집합은 $N = \{S(0), S(1), S(2)\}$이고, 심볼값들의 집합은 $M = \{X(0), X(1), X(2), X(3), X(4)\}$이다.

<288> 또한, 상태 확률이, 표 3에 나타낸 바와 같이, $P_{S(0)} = 0.5,\ P_{S(1)} = 0.2,\ P_{S(2)} = 0.3$인 것으로 가정하자. 따라서, 상태 0에 있을 확률은 50%이고, 상태 1에 있을 확률은 20%이며, 상태 2에 있을 확률은 30%이다.

<289> <표 3> 상태 확률

<290>

$P_{S(0)}$	$P_{S(1)}$	$P_{S(1)}$
0.5	0.2	0.3

<291> 표 4는 상태들 각각에서의 심볼값에 대한 실제 확률 분포$P_{S(j),X(i)}$를 나타낸 것이다.

<292> <표 4> 상태들에서의 심볼값에 대한 실제 확률 분포

<293>

	$X(0)$	$X(1)$	$X(2)$	$X(3)$	$X(4)$
$P_{S(0),X(i)}$	0.09	0.4	0.04	0.4	0.07
$P_{S(1),X(i)}$	0.055	0.7	0.03	0.2	0.015
$P_{S(2),X(i)}$	0.165	0.1	0.09	0.6	0.045

<294> 임의적인 임계값으로서, 상태들 중 임의의 상태에 대해, 그 상태에 있는 심볼값의 확률 × 그 상태에 있을 확률이 0.1보다 클 경우 심볼값 $X(i)$이 확률이 높은 집합 Q에 속하는 것으로 가정하자. 즉, 주어진 $X(i)$에 대한 임의의 $S(j)$에 대해 $P_{S(j),X(i)} * P_{S(j)} > 0.1$인 경우, 심볼값 $X(i)$는 집합 Q에 있다. 그렇지 않은 경우, 심볼값 $X(i)$는 집합 R에 있다. 표 4에서의 분포에 있어서, $L = 2, Q\{X(1), X(3)\}$이고 $R = \{X(0), X(2), X(4)\}$이다. (유의할 점은 $P_{S(2),X(0)} > P_{S(2),X(1)}$이고, 심볼값 $X(1)$이 확률이 높은 심볼값으로 지정되어 있는 반면 심볼값 $X(0)$이 확률이 낮은 심볼값으로 지정되어 있다는 것이다. 상태 S(1)에서, X(1)은 아주 높은 확률을 갖는다.) 다른 대안으로서, 임계값 및/또는 테스트가 서로 다르다. 예를 들어, 임계값이 심볼값들의 비율의 관점에서 설정되거나 테스트가 다수의 서로 다른 상태들에서 높은 확률을 필요로 한다. 일반적으로, 집합 Q 및 R의 크기에 관한 주어진 제약조건에 대해, 실제 분포와 근사 분포 간의 상대 엔트로피(relative entropy)를 살펴봄으로써 최적의 분할(optimal partition)이 구해질 수 있다. (일반적으로, 본 명세서에서 사용되는 바와 같이, "최적의"라는 용어는 어떤 파라미터화 또는 모델링에 따라 다른 해보다 더 나은 어떤 일련의 기준을 만족시키는 해를 말하며, 상황에 따라서는 절대적인 관점에서 최적이거나 최적이 아닐 수 있고, "최적화하다"라는 용어는 이러한 해를 구하는 프로세스를 말하는 데 사용된다.)

<295> 근사화에서, 집합 Q 내의 심볼값 $X(i)$에 대해 $P'_{S(j),X(i)} = P_{S(j),X(i)}$이다. 상태 $S(j)$에 대한 분포는

집합 Q 내의 심볼값에 대해 수정되지 않는다. 그렇지만, 집합 R 내의 심볼값 $X(i)$에 대해서는, 근사 분포가 다르다. 맨 먼저, 집합 R 내의 심볼값들에 대한 실제의 조건부 분포 $P_{S(j),X(i),R}$가 계산된다. 집합 R 내의 심볼값에 대해, 실제의 조건부 분포(집합 Q 내의 심볼값 X(1),X(3)의 기여분을 제거하고 X(0),X(2),X(4)로부터의 기여분만으로 가중함)가 표 5에 주어져 있다. $P_{S(0),X(0),R}$는 0.09 / (0.09 + 0.04 + 0.07) = 0.45이고, $P_{S(0),X(1),R}$는 0.04 / (0.09 + 0.04 + 0.07) = 0.2이다.

<296> <표 5> 집합 R 내의 심볼값에 대한 실제의 조건부 확률

	$X(0)$	$X(2)$	$X(4)$
$P_{S(0),X(i),R}$	0.45	0.2	0.35
$P_{S(1),X(i),R}$	0.55	0.3	0.15
$P_{S(2),X(i),R}$	0.55	0.3	0.15

<297>

<298> 이어서, 근사 조건부 분포 $P'_{S(j),X(i),R}$가 수학식 1과 같이 계산된다.

수학식 1

$$P'_{S(j),X(i),R} = \sum_{S(j)=S(0)}^{S(N-1)} P_{S(j)} * P_{S(j),X(i),R}$$

<299>

<300> 즉, 집합 R에 있을 때의 근사 조건부 분포는 N개의 상태에 걸친 실제 조건부 분포 $P_{S(j),X(i),R}$의 ($P_{S(j)}$에 의해) 가중된 평균이다. 표 4 및 표 5의 값들에 대해, 집합 R에 있을 때의 근사 조건부 분포 $P'_{S(j),X(i),R}$가 표 6에 나타내어져 있다. X(0)에 대해, $P'_{S(j),X(0),R}$는 (0.5 * 0.45) + (0.2 * 0.55) + (0.3 * 0.55) = 0.5이다.

<301> <표 6> 집합 R 내의 심볼값들에 대한 근사 조건부 분포

	$X(0)$	$X(2)$	$X(4)$
$P'_{S(j),X(i),R}$	0.5	0.25	0.25

<302>

<303> 각각의 상태 $S(j)$에 대한 최종 근사 분포는 수학식 2이다.

수학식 2

$$P'_{S(j),X(i)} = \begin{cases} P'_{S(j),X(i),R} * \sum_{X(i)\in R} P_{S(j),X(i)} & X(i) \in R \text{ 인 경우} \\ P_{S(j),X(i)} & X(i) \in Q \text{ 인 경우} \end{cases}$$

<304>

<305> 따라서, 집합 Q 내의 심볼값에 대해, 상태 $S(j)$에서의 실제 확률값이 상태 $S(j)$에 대한 근사 분포에서 사용된다. 집합 R 내의 심볼값에 대해, 심볼값에 대한 근사 조건부 분포 확률 $P'_{S(j),X(i),R}$이 상태 $S(j)$에 대한 집합 R 내의 심볼값들에 대한 실제 확률의 합과 곱해진다. 심볼값 X(0) 및 상태 S(0)에 대해, $P'_{S(0),X(0)}$은 0.5 * (0.09 + 0.04 + 0.07) = 0.1이다. 표 4 및 표 6에서의 다른 값들에 대해, 상태 $S(j)$에 대한 최종 근사

확률 분포는 표 7에 주어져 있다.

<306> <표 7> 상태들에서의 심볼값에 대한 최종 근사 분포

	$X(0)$	$X(1)$	$X(2)$	$X(3)$	$X(4)$
$P_{S(0),X(i)}$	0.1	0.4	0.05	0.4	0.05
$P_{S(1),X(i)}$	0.05	0.7	0.025	0.2	0.025
$P_{S(2),X(i)}$	0.15	0.1	0.075	0.6	0.075

<307>

<308> 기본적으로, 표 7을 표 4와 비교하면, 확률이 높은 심볼값 X(1),X(3)에 대해서는 분포가 그대로이고, 확률이 낮은 심볼값 X(0),X(2),X(4)에 대해서는 집합 R 내의 심볼값들에 대한 상대 확률이 상태마다 동일하다는 조건을 적용하기 위해 분포가 변화되었다. 즉, 표 7에서의 각각의 상태에서, X(0)는 X(2)의 2배 확률이고, X(0)는 X(4)의 2배 확률이다.

<309> 일반적인 경우에, M개의 심볼값에 대한 N개의 상태에서 시작하여, 심볼값들(집합 R) 중 어떤 것에 대한 상태의 수가 집합 R에 대한 N개의 조건부 분포를 P개(단, P < N임)의 분포로 클러스터링함으로써 감소될 수 있다. 이 절차는 이어서 M개의 심볼값의 어떤 다른 부분집합에 대해 반복될 수 있다. 이는 또한 집합 R의 P개의 클러스터링된 분포에 대해 재귀적으로 반복될 수 있으며, 여기서 집합 R은 P개의 상태를 갖는 $|R|$개의 심볼값($|R|$은 카디날리티(cardinality), 즉 집합 R 내의 원소의 수를 나타냄)을 갖는다. 이것은 M개의 심볼값에 대한 N개의 상태(또는 분포, 또는 클러스터)에 제약조건을 부과한다. 이들 제약조건은 M개의 심볼값에 대한 N개의 상태가 고정된 후에 적용될 수 있거나, 더 최적으로는, 훈련 단계 자체 동안에 적용될 수 있다. 이 훈련은 M개의 심볼값에 대한 많은 수의 분포로 시작하고, 그 결과 이들이 조건부 분포에 대한 추가의 제약조건을 만족시키도록 하는 N개의 클러스터링된 분포가 얻어진다.

<310> **2. 예시적인 VLC 테이블**

<311> 허프만 코딩 및 디코딩 및 다른 가변 길이 코딩 및 디코딩을 비롯한 다양한 유형의 적응적 엔트로피 코딩 및 디코딩에서, 서로 다른 상태에서의 심볼값들에 대한 근사 분포가 사용될 수 있다.

<312> 허프만 코드 테이블은 트리로 생각될 수 있고, 이 트리에서의 각각의 리프(leaf)는 심볼값에 대응한다. 트리의 좌측 가지는 하나의 이진값(예를 들어, 0)과 연관관계를 가지며, 트리의 오른쪽 가지는 반대 이진값(예를 들어, 1)과 연관관계를 갖는다. 도 23에 도시된 트리는 표 7에 나타낸 근사 분포에 대응한다.

<313> 도 23에서, 각자의 트리의 점선 부분은 집합 R 내의 심볼값들에 대한 것이고, 트리의 다른 부분은 집합 Q 내의 심볼값들에 대한 것이다. 표 7에 나타낸 근사 분포에서, 집합 R 내의 심볼값의 조건부 분포가 상태에 상관없이 동일하며, 따라서 도 23에서의 각각의 트리는 집합 R 내의 심볼값에 대해 공통의 동일한 가지를 가질 수 있다. 공통의 동일한 가지의 배치는 트리에서 아무 곳에나 있을 수 있으며, 일반적으로 공통의 가지에 표현된 심볼값의 확률의 누계가 트리의 다른 심볼값의 확률과 어떻게 비교되는지에 의존한다. 따라서, 공통의 가지는 트리마다 더 위쪽에 또는 더 아래쪽에 있을 수 있다.

<314> 도 23의 주어진 트리/상태에 대해, 집합 R 내의 모든 심볼값에 대한 VLC는 트리에서의 가지의 배치에 의해 표시된 것과 동일한 프리픽스(prefix)를 갖는다. 그에 부가하여, 도 23에서의 상태와 상관없이, 집합 R 내의 각각의 심볼값은 공통의 동일한 가지에 의해 표시된 것과 동일한 공통의 서픽스(suffix)를 갖는다. 도 23의 트리에서, 예시적인 허프만 코드는 다음과 같다.

<315> <표 8> 예시적인 허프만 코드 및 테이블

<316>

	S(0)에 대한 허프만 코드	S(1)에 대한 허프만 코드	S(2)에 대한 허프만 코드
X(0)	110	110	100
X(1)	0	0	11
X(2)	1110	1110	1010
X(3)	10	10	0
X(4)	1111	1111	1011

<317> 동일한 테이블이 상태 S(0) 및 S(1)에 대해 사용될 수 있다. 상태 S(0) 및 S(1)에서, 집합 R 내의 심볼값에 대한 공통의 프리픽스(밑줄로 표시됨)은 집합 R 내의 심볼값에 상관없이 "11"이다. 상태 S(2)에서, 집합 R 내의 심볼값에 대한 공통의 프리픽스(밑줄로 표시됨)는 "10"이다. 상태 S(0), S(1) 및 S(2)에서, 각자의 심볼값에 대한 서픽스(굵은체로 표시됨)는 동일하다. (X(0)에 대한 서픽스는 "0"이고, X(1)에 대한 서픽스는 "10"이며, X(2)에 대한 서픽스는 "11"이다)

<318> 이 경우에, 근사화된 분포에 대한 허프만 코드가 집합 R 내의 심볼값들에 대한 2-단계 코딩/디코딩을 용이하게 해주고 또 이를 사용하여 구현될 수 있다. 표 8에 나타낸 코드는 표 9 및 표 10에 나타낸 바와 같이 추가적으로 분할될 수 있다.

<319> <표 9> 각자의 상태에 대한 첫번째 단계 코드 테이블

<320>

	S(0)에 대한 허프만 코드	S(1)에 대한 허프만 코드	S(2)에 대한 허프만 코드
X(1)	0	0	11
X(3)	10	10	0
X(0), X(2), X(4)	11	11	10

<321> <표 10> 모든 상태들에 대한 두번째 단계 코드

<322>

	S(0), S(1) 및 S(2)에 대한 허프만 코드
X(0)	0
X(2)	10
X(4)	11

<323> 집합 R 내의 값을 갖는 심볼에 대해, 인코더는 먼저 집합 R 내의 심볼값들 전부를 나타내는 이스케이프 코드를 코딩한다. 이것은 특정의 상태에 대한 집합 Q 내의 심볼값들에 대한 제1 코드 테이블로부터 모든 상태들에 걸쳐 집합 R 내의 심볼값들에 대한 제2 코드 테이블로의 전환을 신호한다. 인코더는 이어서 제2 코드 테이블로부터 적절한 코드를 코딩한다.

<324> 허프만 코드 테이블의 더 복잡한 계층적 구성에서, 허프만 코드 테이블은 다수의 공통 가지를 포함할 수 있으며, 각각의 공통 가지는 심볼값들의 서로 다른 부분집합에 대한 단일의 조건부 분포에 대응한다. 2-단계 구현에서, 첫번째 단계 허프만 코드 테이블은, 다수의 공통 가지들 각각에 대해 하나씩, 다수의 이스케이프 코드를 포함할 수 있다.

<325> 보다 일반적으로, 허프만 코드 테이블은 임의적인 계층구조로 구성될 수 있으며, 이스케이프 코드(및 가능한 다른 선택 정보)는 다른 허프만 코드 테이블 또는 일련의 허프만 코드 테이블로 전환하는 데 사용된다.

<326> 특정의 테이블에서, 이스케이프 코드는 또한 (다른 테이블로 전환하기 보다는) 어떤 심볼값들에 대한 고정 길이 코딩/디코딩 방식으로 전환하는 데 사용될 수 있다.

<327> 다른 대안으로서, 허프만 코드의 규칙을 따르지 않는 다른 유형의 VLC 테이블이 구성된다. 예를 들어, 단일의 VLC 테이블이 VLC를 일군의 상태들 전부에 대한 집합 R 내의 심볼값들과 연관시키고, 다수의 VLC 테이블(그룹의 상태마다 하나의 테이블)이 VLC를 집합 Q 내의 심볼값들과 연관시킨다.

<328> 게다가, 이전의 예들이 고정된, 사전-훈련된 코드 테이블을 나타내고 있지만, 다른 대안으로서, 코드 테이블은 처리된 심볼값들에 따라 그의 코드를 동적으로 변화시킨다. 이러한 동적으로 변하는 테이블의 경우, 인코더 및 디코더는 여전히 어떤 심볼값들에 대해서는 다수의 코드 테이블을, 다른 심볼값들에 대해서는 단일의 코드 테이블을 선택적으로 사용할 수 있다.

<329> 일반적으로, M개의 심볼값에 대해 N개의 상태가 있는 경우, N개의 VLC 테이블이 있거나 또는 허프만 코드를 사용하는 경우 N개의 트리가 있다. M개의 심볼값의 L개의 서로 소인 부분집합이 있는 경우(L개의 부분집합 각각은 P_l개 상태(단, l = 0, 1,...,L-1이고 모든 l에 대해 P_l<N임)를 가짐), N개의 트리 각각은 L개의 가지($b_0, b_1, ..., b_{L-1}$로 표시됨)를 가지며, 각각의 가지 b_l는 그 부분집합 l에 대해 이용가능한 P_l개 공통 가지 중

하나로부터 선택된다. 게다가, L개의 부분집합 중 임의의 부분집합이 다시 부분집합으로 재귀적으로 분할되는 경우(각각의 부분집합은 그의 부모 집합보다 더 적은 상태를 가짐), P_l 개 가지 중의 가지들에 대해 마찬가지로 말해질 수 있다.

<330> **3. 산술 코딩/디코딩에 대한 예시적인 분포**

<331> 다른 인코더/디코더에서는, 산술 코딩/디코딩에 근사 분포가 사용된다. 산술 코딩은 일반적으로 일련의 심볼을 주어진 범위 내의 하나의 수로 표현하는 것을 포함한다. 일반적으로, 이 수는 0과 1 사이의 소수이다. 심볼은 그 심볼을 범위의 일부 내에 놓음으로써 코딩되며, 그 범위는 심볼값의 확률 분포에 따라 분할된다.

<332> 산술 코딩 및 디코딩에서 사용하기 위해, 표 7에 나타낸 근사 분포는 표 6 및 표 11로 분할될 수 있다. X(0), X(2) 및 X(4)에 대한 표 11에서의 전환값은 표 11에 나타낸 상태/분포 중 하나로부터 표 6에 나타낸 상태/분포로의 변경을 나타낸다.

<333> <표 11> Q 내의 심볼값들이 합성된 근사 분포

<334>

	$X(1)$	$X(3)$	$X(0), X(2), X(4)$
$P_{S(0),X(i)}$	0.4	0.4	0.2
$P_{S(1),X(i)}$	0.7	0.2	0.1
$P_{S(2),X(i)}$	0.1	0.6	0.3

<335> 이전의 예가 고정된, 사전-훈련된 분포를 나타내고 있지만, 다른 대안으로서, 분포는 처리된 심볼값들에 따라 동적으로 변한다. 이러한 동적으로 변하는 분포의 경우, 인코더 및 디코더는 여전히 어던 심볼값들에 대해서는 다수의 분포를, 다른 심볼값들에 대해서는 단일의 분포를 선택적으로 사용할 수 있다.

<336> **4. 엔트로피 모델을 결정하는 예시적인 훈련**

<337> 인코더 및 디코더가 심볼들에 대한 다수의 엔트로피 모델을 선택적으로 사용할 때, 엔트로피 모델은 궁극적으로 심볼들에 대한 확률 분포 정보에 의존한다. 어떤 구현에서, 인코더 또는 통계 분석 소프트웨어 등의 도구는 엔트로피 모델에 대한 상태 및 확률 분포를 결정하기 위해 이하의 접근방법을 사용한다.

<338> 도 24는 다중 엔트로피 모델 코딩/디코딩 방식에 있어서 확률 분포를 상태들로 클러스터링하는 2-단계 기법(2400)을 나타낸 것이다. 이 기법(2400)은 심볼값들의 확률 분포를 훈련 벡터로서 취급하고, 이 훈련 벡터는, 벡터 양자화 방식에 사용되는 클러스터링 접근방법과 유사하게, 클러스터들로 그룹화된다.

<339> 맨 먼저, 이 도구는 훈련 벡터에 대한 실제 확률 분포를 구한다(2410). 이 훈련 벡터는 대표적인 소스들의 훈련 세트로부터 온 것이다. 예를 들어, 오디오 코딩/디코딩의 경우, 서브-프레임 내의 심볼값들의 확률 분포가 하나의 훈련 벡터로 된다. 일반적인 오디오 코딩/디코딩의 경우, 확률 분포가 서로 다른 오디오 소스의 다수의 서브-프레임으로부터 얻어지도록, 훈련 세트가 다수의 오디오 소스를 포함한다. 다양한 비트 레이트 및/또는 품질 설정에서 훈련하는 것으로부터 훈련 벡터가 얻어질 수 있다.

<340> 이 도구는 이어서 제1 비용 메트릭(cost metric)을 사용하여 훈련 벡터를 클러스터링한다(2420). 예를 들어, 제1 비용 메트릭은 MSE(mean squared error, 평균 제곱 오차)이다. 클러스터링 자체는 도 25를 참조하여 설명하는 바와 같이 GLA(generalized Lloyd algorithm)의 변동을 사용하거나 어떤 다른 메카니즘을 사용할 수 있다. 기본적으로, GLA 변동에서, 이 도구는 훈련 벡터들을 주어진 수의 클러스터로 반복하여 클러스터링하며, 주어진 디코더에 대한 최적의 인코더를 찾는 것과 주어진 인코더에 대한 최적의 디코더를 찾는 것 사이에서 반복한다. 어떤 수의 반복 이후에, 이 도구는 제1 비용 메트릭이 최소화되도록 하는 일련의 클러스터를 찾아낸다.

<341> 이 도구는 이어서 제2 비용 메트릭을 사용하여 클러스터들을 세분(refine)한다(2430). Itakura-Saito 거리는 2개의 확률 분포 간의 상대 엔트로피를 측정하는 한가지 방법이다. 세분(2430)에서, 클러스터링 로직의 일부가 제1 비용 메트릭에서 사용되는 클러스터링 로직의 일부와 동일하거나 그와 다를 수 있다.

<342> 따라서, 도 24에 따르면, 이 도구는 2-단계 훈련 프로세스를 사용한다. 첫번째 단계에서, 이 도구는 제1 비용 메트릭(예를 들어, MSE)을 사용하여 분포들에 대한 근사 PMF(probability mass function, 확률 질량 함수) 클러스터를 얻는다. 두번째 단계에서, 이 도구는 제2 비용 메트릭(예를 들어, Itakura-Saito 거리)을 사용하여 PMF 클러스터를 추가적으로 세분한다. MSE는 계산하기가 비교적 간단하지만, 코딩/디코딩을 위해 엔트로피는

물론 상대 엔트로피 메트릭도 모델링하지 않는다. 반면에, 상대 엔트로피는 클러스터를 세분하기 위한 효과적인 메트릭이지만, 그것이 사용되는 유일한 메트릭일 때, 그 결과 최적이 아닌 클러스터링이 얻어질 수 있다. 많은 경우에, 2-단계 훈련은 복잡도의 관점에서 더 빠를 뿐만 아니라(왜냐하면 상대 엔트로피가 계산하기가 더 복잡하기 때문임) 그 결과 코딩/디코딩 응용에 대한 더 나은 클러스터도 얻어진다.

<343> 다른 대안으로서, 도구는 상태 및 확률 분포를 결정하는 다른 접근방법을 사용한다. 예를 들어, 이 도구는 첫 번째 또는 두번째 비용 메트릭에 대해 MSE 또는 상대 엔트로피가 아닌 메트릭을 사용한다. 또는, 이 도구는 1-단계 프로세스에서 하나의 비용 메트릭을 사용한다.

<344> 도 25는 GLA의 변형에 따라 훈련 벡터를 클러스터링하는 기법(2500)을 나타낸 것이다. 도 24에서와 같이, 이 기법(2500)은 심볼값의 확률 분포를 훈련 벡터로서 취급하고, 훈련 벡터들이 클러스터로 그룹화된다.

<345> 맨 먼저, 이 도구는 훈련 벡터들로부터 하나의 클러스터를 계산한다(2510). 예를 들어, 일반적인 오디오 코딩/디코딩의 경우, 훈련 벡터는 서로 다른 비트 레이트 및/또는 품질 설정에서 인코딩된 오디오 파일 등의 서로 다른 오디오 소스로부터의 서브-프레임에 대한 확률 분포이다. 얻어진 훈련 벡터의 수는 구현에 따라 다르다. 한 구현에서, 이 도구는 계산된 최종 클러스터보다 약 100배 더 많은 훈련 벡터를 얻는다. 하나의 클러스터는, 훈련 벡터들 또는 훈련 벡터들의 어떤 다른 조합을 평균함으로써 계산된, 훈련 벡터들의 무게 중심(centroid)이다.

<346> 이 도구는 이어서 하나의 클러스터를 다수의 클러스터로 분할한다(2520). 예를 들어, 이 도구는 주성분 분석(principal component analysis)을 사용하여 하나의 클러스터를 2개의 클러스터로 분할하고, 그 하나가 원래의 클러스터이고, 다른 하나가 [원래의 클러스터 + 구현-의존적인 상수 × 주성분](예를 들어, 다른 하나는 주성분의 방향을 따라 어떤 오프셋에 있는 클러스터임)이다. 다른 대안으로서, 이 도구는 어떤 다른 분석을 사용하여 클러스터를 다수의 클러스터로 분할한다.

<347> 이 도구는 어떤 비용 메트릭에 따라 훈련 벡터를 다수의 현재 클러스터로 분류한다(2530). 예를 들어, 비용 메트릭은 MSE, 상대 엔트로피, 또는 어떤 다른 메트릭이다. 훈련 벡터 대 클러스터의 MSE는 훈련 벡터의 확률 분포점과 클러스터의 대응하는 점 간의 유클리드 거리를 나타낸다. 훈련 벡터와 클러스터 간의 상대 엔트로피는 다음과 같이 훈련 벡터와 클러스터 간의 차이를 제공한다.

수학식 3

<348>
$$-\sum_{k} training\_vector_k * \log_2(cluster_k)$$

<349> 여기서, k는 훈련 벡터 및 클러스터에서의 점을 가리킨다. 형식에 덜 구애되어, 상대 엔트로피는 훈련 벡터와 클러스터 간의 불일치로 인한 비트 레이트 불이익을 나타낸다. 이 도구는 훈련 벡터가 가장 낮은 MSE, 가장 낮은 상대 엔트로피, 기타 등등을 갖는 클러스터로 훈련 벡터를 분류한다.

<350> 이 도구는 분류된 훈련 벡터로부터 현재의 클러스터를 재계산한다(2540). 예를 들어, 각각의 현재의 클러스터에 대해, 이 도구는 그 클러스터로 분류된 훈련 벡터들의 무게중심을 계산한다. 다른 대안으로서, 이 도구는 각각의 현재의 클러스터를 그 클러스터로 분류된 훈련 벡터의 어떤 다른 조합으로서 계산한다.

<351> 이 도구는 클러스터가 안정화되었는지를 판정한다(2545). 예를 들어, 이 도구는 재계산(2540) 이전 대 그 이후의 클러스터의 변화가 어떤 기준을 만족시키는지를 검사한다. 한 기준은 클러스터가 재계산(2540)에서의 어떤 임계값량보다 더 시프트되지 않은 것이며, 여기서 이 임계값량은 구현에 의존한다. 다른 대안으로서, 이 도구는 다른 및/또는 부가적인 기준을 고려한다. 클러스터가 안정화되지 않은 경우, 이 도구는 비용 메트릭에 따라 (재계산(2540)된) 현재의 클러스터들 간에 훈련 벡터를 분류한다.

<352> 현재의 클러스터가 안정화되었을 때, 이 도구는 충분한 클러스터가 있는지를 판정한다(2550). 일반적으로, 클러스터의 원하는 수는 메모리 사용 대 인코딩 성능을 절충하도록 설정될 수 있다. 더 많은 클러스터를 가지면, 분포, VLC 테이블, 기타 등등을 저장하기 위한 메모리 사용의 증가의 대가로, 엔트로피 모델에 더 많은 상태 및 적응성을 가져오는 경향이 있다. 전방 적응이 사용될 때, 더 많은 클러스터를 갖는 것은 또한 (분포, 테이블, 기타 등등을 알려주기 위해) 더 많은 부수 정보가 신호된다는 것을 의미한다. 이와 반대로, 더 적은 클러스터를 가지면, 훈련 벡터와 최종 클러스터 간의 불일치를 증가시키는 경향이 있으며, 이는 보통 인코딩 동안에 엔트로피 모델과 심볼값의 실제 분포 간의 불일치의 증가를 나타낸다.

<353> 클러스터의 원하는 수에 도달하지 못한 경우, 이 도구는 현재의 클러스터의 일부 또는 그 전부를 분할한다(2560). 예를 들어, 이 도구는 주성분 분석 또는 어떤 다른 분석을 사용하여 클러스터를 2개의 클러스터로 분할한다. 이 도구가 G개의 최종 클러스터를 구하고 있고 지금 F개(단, F < G임)의 현재 클러스터를 갖는 것으로 가정하자. F개의 현재 클러스터 각각을 분할하는 결과 너무 많은 클러스터가 얻어지는 경우, 이 도구는 G - F개의 상위 현재 클러스터(예를 들어, 몇개의 훈련 벡터가 현재의 클러스터로 어떻게 분류되는지의 관점에서의 "상위") 각각을 2개의 클러스터로 분할할 수 있다. 또는, 이 도구는 단순히 각각의 반복에서 상위 클러스터를 분할하거나 분할을 위해 어떤 다른 규칙을 사용할 수 있다. 이 도구는 이어서 비용 메트릭에 따라 (분할(2560)된) 현재의 클러스터들 간에 훈련 벡터를 분류한다(2530).

<354> 현재의 클러스터가 안정화되고 클러스터의 원하는 수에 도달했을 때, 기법(2500)이 종료된다. 분류(2530), 재계산(2540), 및 분할(2560)은 본질적으로 GLA 변형의 반복을 구성하고, 이 반복 동안에, 비용 메트릭이 감소된다.

<355> 도 25의 기법(2500)은 다음과 같이 도 24의 기법에 포함될 수 있다. 이 도구는 클러스터의 원하는 수에 도달될 때까지 비용 메트릭으로서 MSE를 사용하여 도 25의 기법을 수행한다. 그 때에, 이 도구는 클러스터가 안정화될 때까지/어떤 임계값량 이상으로 시프트하지 않을 때까지 상대 엔트로피를 사용하여 분류(2530), 재계산(2540) 및 안정성 검사(2545)를 반복하여 수행한다.

<356> 기법(2400, 2500)은 실제 분포에 근사하지만 어떤 심볼값에 대해 동일한 조건부 분포를 갖는 확률 분포로 최종 클러스터를 생성하는 데 사용될 수 있다. 섹션 V.A.1의 분석 프레임워크의 관점에서, 기법(2400, 2500)은, 분류 및 클러스터링 동작에서, 집합 R 내의 심볼값에 대한 조건부 분포가 모든 클러스터/상태에 대해 동일해야 한다는($P'_{S(j),X(i),R}$ 이 모든 상태 $S(j)$에 대해 동일해야 한다는) 제약조건을 추가함으로써, 표 7에 나타낸 것과 같은 근사 확률 분포를 생성하는 데 사용될 수 있다. 본질적으로, 집합 R 내의 심볼값에 대응하는 클러스터의 그 크기들은 수학식 1 및 2에 나타낸 바와 같이 제약된다. 이 분석에서, 주어진 상태에 있을 확률 $P_{S(j)}$는 그 상태에 대한 클러스터로 분류된 훈련 벡터의 수로 표시된다. 다른 제약조건은 클러스터들 각각의 크기의 합이 1이라는 것이다.

<357> 도 25를 참조하면, 현재의 클러스터의 재계산(2540) 이후에, 하나 이상의 조건부 분포 제약조건이 부과될 수 있다. 일반적으로, M개의 심볼값에 대해 N개의 상태가 있고 또 M개의 심볼값의 L개의 부분집합이 있으며, L개의 부분집합 각각이 P_l개의 상태(단, $P_l < N, l=0, 1, \ldots, L\text{-}1$임) 및 E_l개의 요소를 갖는 것으로 가정하자. L개의 부분집합 중 주어진 하나의 부분집합 내의 심볼값들 전부가 공통의 (이스케이프/전환) 심볼값으로 그룹화될 수 있다. L개의 이러한 이스케이프/전환 심볼값이 있다. 이어서, 훈련은 $M - (E_0 + E_1 + \ldots + E_{L\text{-}1}) + L$개 심볼값에 대한 N개의 클러스터(또는 분포)를 계속하여 찾아낸다(L개의 부분집합 내의 E_l개의 요소를 제거하고 이스케이프/전환 심볼값에 대한 L개의 요소를 추가함). 이어서, M개의 심볼값의 L개의 부분집합 각각에 대해, 그 부분집합 내에서 조건부 분포(들)가 계산된다. 이 훈련은 이들 부분집합 각각에 대해 P_l개(단, $l = 0, 1, \ldots, L\text{-}1$임)의 클러스터를 찾아내기 위해 L개의 부분집합 각각에 대해 반복된다. 이것을 위한 훈련 벡터는 각각 L개의 부분집합 내의 조건부 분포(들)이다. L개의 부분집합 중 임의의 부분집합이 추가적으로 세분되는 경우, 이 절차는 그 세분된 부분집합 l에 대해 재귀적으로 반복될 수 있는데, 그 이유는 E_l개의 심볼값에 대해 이제 P_l개의 상태가 있기 때문이다.

<358> 어느 심볼값이 집합 Q 및 R 내에 있는지를 지정하는 것에 관해서, 처음에 이것은 하나의 시작 클러스터의 확률 분포에 기초하고 있다. 그 다음에, 집합 Q 및 R의 구성요소는 각각의 상태에 있을 확률(각각의 클러스터에서의 훈련 벡터의 비율) 및 클러스터에 대한 확률 분포에 의존한다.

<359> **5. 대안들**

<360> 이전의 예들 대부분은 어떤 심볼값에 대해 다수의 분포/테이블을 사용하는 것 및 다른 심볼값에 대해 하나의 분포/테이블을 사용하는 것을 수반한다. 이 구성이 일반적으로 엔트로피 코딩 성능을 그다지 손상시키지 않고 메모리 사용을 감소시키고 있지만, 섹션 V에 기술된 기법 및 도구는 더 일반적으로 계층적으로 구성된 엔트로피

모델에 적용가능하다. 인코더 또는 디코더는 선택적으로 어떤 심볼값에 대해서는 더 많은 적응성을 가능하게 해주고 다른 심볼값에 대해서는 더 적은 적응성을 가능하게 해주는 계층적 구조에서의 서로 다른 엔트로피 모델 중에서 선택할 수 있다.

<361> 계층적으로 구성된 엔트로피 모델은 전환마다 다수의 엔트로피 모델을 참조할 수 있다(예를 들어, 단지 확률이 낮은 심볼값에 대한 하나의 엔트로피 모델로 전환하는 것만이 아님). 예를 들어, 어떤 레벨에 있는 일련의 허프만 코드 테이블은 하나의 허프만 코드 테이블 또는 다수의 허프만 코드 테이블을 포함한다. 훈련은 다수의 단계로 행해질 수 있다. 첫번째 훈련 단계에서, 심볼값은 집합 Q 또는 집합 R에 있는 것으로 지정되며, 여기서 집합 R 내의 심볼값에 대한 조건부 분포는 모든 상태에 대해 동일하다. 이어서, 집합 R 내의 심볼값에 대한 차후의 훈련 단계에서, 집합 R 내의 심볼값에 대한 조건부 분포에 대한 이전의 제약조건이 제거되고, 집합 R 내의 심볼값에 대한 확률 분포가 서로 다른 엔트로피 모델에 대한 다수의 클러스터/상태로 분류된다.

<362> 엔트로피 모델 세트의 각각의 멤버는 다른 레벨에 있는 다른 엔트로피 모델 세트로의 다수의 전환점을 포함할 수 있다. 예를 들어, 전방 적응의 경우, 허프만 코드 테이블의 제1 세트의 각각의 테이블은 2개의 이스케이프 코드, 즉 하나 이상의 허프만 코드 테이블의 제2 세트로의 제1 이스케이프 코드 및 하나 이상의 허프만 코드 테이블의 제3 세트로의 제2 이스케이프 코드를 포함한다. 훈련에 관해서는, 심볼값이 엔트로피 모델의 제1 세트에 대해서는 집합 Q에 있거나, 엔트로피 모델의 제2 세트에 대해서는 집합 R에 있거나, 엔트로피 모델의 제3 세트에 대해서는 집합 S에 있는 것으로 지정될 수 있다. 집합 R 내의 심볼값(Q 및 S 내의 심볼값을 무시함)에 대한 조건부 분포는 모든 상태에 대해 동일하며, 집합 S 내의 심볼값(Q 및 R 내의 심볼값을 무시함)에 대한 조건부 분포는 모든 상태에 대해 동일하다.

<363> 부가적인 폭은 별도로 하고, 계층적으로 구성된 엔트로피 모델은 3, 4, 또는 그 이상의 레벨의 엔트로피 모델을 포함할 수 있다. 예를 들어, 전방 적응의 경우, 허프만 코드 테이블의 제1 세트의 각각의 테이블은 허프만 코드 테이블의 제2 세트로의 이스케이프 코드를 포함하고, 허프만 코드 테이블의 제2 세트의 각각의 테이블은 허프만 코드 테이블의 제3 세트로의 이스케이프 코드를 포함한다. 훈련은 다수의 단계로 행해질 수 있다. 첫번째 단계에서, 심볼값이 엔트로피 모델의 제1 세트에 대해서는 집합 Q에 있는 것으로 지정되거나 엔트로피 모델의 다른 세트들에 대해서는 집합 R에 있는 것으로 지정된다. 집합 R 내의 심볼값(Q 내의 심볼값을 무시함)에 대한 조건부 분포는 모든 상태에 대해 동일하다. 이어서, 집합 R 내의 심볼값에 대한 부가적인 훈련 단계에서, 조건부 분포에 대한 이 제약조건은 제거되고, 집합 R 내의 심볼값이 엔트로피 모델의 제2 세트에 대해서는 집합 S에 있는 것으로 지정되거나 엔트로피 모델의 임의의 다른 세트에 대해서는 집합 T에 있는 것으로 지정된다. 이 단계에서, 집합 T 내의 심볼값(S 내의 심볼값을 무시함)에 대한 조건부 분포는 모든 상태에 대해 동일하다.

<364> 가변 길이(예를 들어, 허프만) 코딩 및 디코딩과 산술 코딩 및 디코딩은 별도로 하고, 다른 유형의 엔트로피 코딩 및 디코딩은 엔트로피 모델의 선택적 사용을 포함할 수 있다. 예를 들어, 가변 길이 대 가변 길이(variable to variable) 인코딩 및 디코딩은 계층적 구조로 VLC 테이블을 포함할 수 있다.

<365> **C. 예시적인 인코딩 기법**

<366> 도 26은 다수의 엔트로피 모델의 선택적 사용으로 심볼을 인코딩하는 기법(2600)을 나타낸 것이다. 도 2, 도 4 또는 도 6에 도시된 인코더와 같은 인코더가 기법(2600)을 수행한다.

<367> 파형 오디오 인코더에서, 심볼들은 일반적으로 양자화된 스펙트럼 계수에 대한 것이다. 양자화된 스펙트럼 계수는 (예를 들어, 계수 예측 또는 계수 재정렬에 의해) 전처리될 수 있다. 심볼들 각각은 양자화된 스펙트럼 계수를 나타낼 수 있다. 또는, 심볼들 각각은 일군의 양자화된 스펙트럼 계수를 나타낼 수 있다. 벡터 허프만 코딩의 경우, 심볼은, 예를 들어, 일군의 4개의 양자화된 스펙트럼 계수를 나타낸다. 런-레벨 코딩의 경우, 심볼은, 예를 들어, 런-레벨 쌍을 나타낸다.

<368> 일련의 심볼에 대해, 인코더는 엔트로피 심볼의 제1 세트로부터 엔트로피 모델을 선택한다(2610). 예를 들어, 인코더는 벡터 허프만 코딩 또는 런-레벨 코딩에 대한 다수의 이용가능한 허프만 코드 테이블 중에서 허프만 코드 테이블을 선택한다. 다른 대안으로서, 인코더는 다른 엔트로피 인코딩 방식에서 사용되는 엔트로피 모델을 선택한다. 어떤 구현들에서, 인코더는 컨텍스트 정보에 따라 엔트로피 모델을 선택한다. 다른 구현들에서, 인코더는 다양한 엔트로피 모델을 사용하여 인코딩의 성능을 평가한 후에 엔트로피 모델을 선택한다. 트렐리스 구조를 사용하는 허프만 코드 테이블에 대한 선택 프로세스의 한 예에 대해 이하에서 기술한다. 다른 대안으로서, 인코더는 다른 메카니즘을 사용하여 엔트로피 모델을 선택한다.

<369> 도 26으로 돌아가서, 인코더는 선택에 따라서는 선택된 엔트로피 모델을 나타내는 정보를 신호한다(2620). 전

방 적응의 경우, 인코더는 선택된 엔트로피 모델을 나타내는 정보를 명시적으로 신호한다. 허프만 코드 테이블 전환에 대한 한가지 전방 적응 메카니즘에 대해 이하에서 상세히 기술한다. 다른 대안으로서, 인코더는 다른 시그널링 메카니즘을 사용한다. 후방 적응의 경우, 엔트로피 모델의 선택은 디코더에서 이용가능한 컨텍스트로부터 추론된다.

<370> 인코더는 이어서 선택된 엔트로피 모델을 사용하여 일련의 심볼을 인코딩한다(2630). 엔트로피 모델에서의 임의의 전환점에서, 인코더는 하나 이상의 엔트로피 모델의 다른 세트로 전환할 수 있다. 예를 들어, 인코더는 제1 허프만 코드 테이블 내의 이스케이프 코드를 사용하여 제2 허프만 코드 테이블로의 전환을 신호하고, 이어서 제2 허프만 코드 테이블을 사용하여 심볼을 인코딩한다.

<371> 인코더는 이어서 엔트로피 코딩된 심볼을 신호한다(2640). 임의의 전환이 일어났을 때, 인코더는 모델 세트 내에서의 선택을 위한 이스케이프 코드 또는 다른 모델 전환 정보 등의 전환 정보도 신호할 수 있다.

<372> 인코더는 그 다음 계열로 계속할지를 판정하고(2650), '예'인 경우, 그 다음 계열의 심볼에 대한 엔트로피 모델을 선택한다(2610). 예를 들어, 한 구현에서 허프만 코드 테이블을 사용하여 양자화된 스펙트럼 계수를 인코딩할 때, 인코더는 바크 경계(bark boundary)에서 코드 테이블을 변경할 수 있다. 환언하면, 주파수 스펙트럼을 분할하는 바크 경계는 제1 코드 테이블 세트로부터 선택된 허프만 코드 테이블을 변경하기 위한 가능한 변경 위치로서 기능한다. 인코딩 중인 현재 심볼에 대한 계수가 바크 경계를 넘어서는 경우(예를 들어, 왜냐하면 심볼이 경계를 가로지르는 계수들의 벡터 또는 계수들의 런-레벨 쌍을 나타내기 때문에), 현재의 심볼의 계수의 끝이 유효한 변경 위치로 된다. 다른 대안으로서, 인코더는 다른 변경 위치에서 제1 모델 세트로부터 엔트로피 모델의 선택을 변경하고, 선택된 엔트로피 모델에 따라 인코딩된 일련의 심볼들이 어떤 다른 기간을 갖는다.

<373> 상기한 바와 같이, 한 구현에서, 인코더는 서로 다른 테이블의 평가를 위해 트렐리스 구조를 사용하는 허프만 코드 테이블을 선택한다. 인코더는 가능한 테이블 전부를 갖는 2개의 유효한 테이블 변경 위치(이들이 바크 경계임) 사이에 심볼들 전부를 인코딩한다. 인코더는 심볼을 인코딩하기 위해 테이블마다 사용되는 비트의 수를 추적한다. 인코더는 최상의 가능한 인코딩을 찾기 위해 트렐리스를 구성하고, 테이블이 변경되는 경우 신호될 비트를 고려한다.

<374> $b_{t,i}$가 테이블 변경 위치 t까지 인코딩할 때 사용되는 최대 비트수이고, 테이블 i가 사용되는 마지막 테이블이라고 가정하자. 비트 카운트 $r_{t,i}$는 테이블 i를 사용하여 변경 위치 i와 변경 위치 t+1 사이에 심볼을 인코딩하는 데 필요한 비트이다. 비트 카운트 $s_{t,i,k}$는 변경 위치 t에서 테이블 i부터 테이블 k까지 테이블 변경을 인코딩하는 데 필요한 비트이다. 환언하면, 변경 위치 t에서 사용 중인 마지막 테이블이 테이블 i이었고, 테이블 k는 이제 변경 위치 t+1까지 인코딩하는 데 사용된다. 테이블 $n_{t,i}$는 변경 위치 t에 있는 현재 테이블이 테이블 i인 최적의 인코딩을 얻기 위해 변경 위치 t-1에서 사용되는 테이블이다. 그러면,

수학식 4

<375>
$$n_{t+1,i} = \arg\min_{k}(b_{t,k} + r_{t,i} + s_{t,i,k})$$
$$b_{t+1,i} = \min_{k}(b_{t,k} + r_{t,i} + s_{t,i,k})$$

<376> 인코더는 $b_{tmax,i}$를 최소로 하는 i를 찾아냄으로써 전체 서브-프레임 또는 시퀀스의 다른 부분에 대한 최적의 인코딩을 결정하며, 여기서 tmax는 t에 대한 최대값이다. 인코더는 n의 값을 조사하여 최적의 경로를 추적함으로써 최적의 테이블을 찾아낸다. 테이블 변경을 코딩하는 데 필요한 비트는 본질적으로 $\log_2$(테이블의 수) + $\log_2$(남은 바크의 수) + 1이다. 테이블이 변경될 때, 인코더는 이것이 사용된 마지막 테이블인지를 알려주기 위해 1 비트를 신호하고, 사용된 마지막 테이블이 아닌 경우, 인코더는 테이블이 몇개의 바크 대역(bark band)에 적용되는지를 인코딩하기 위해 log(남은 바크의 수)를 신호한다.

<377> **D. 예시적인 디코딩 기법**

<378> 도 27은 다수의 엔트로피 모델의 선택적 사용으로 심볼을 디코딩하는 기법(2700)을 나타낸 것이다. 도 3, 도 5 또는 도 7에 도시된 디코더 등의 디코더가 기법(2700)을 수행한다.

<379> 파형 오디오 디코더에서, 심볼은 일반적으로 양자화된 스펙트럼 계수에 대한 것이다. 양자화된 스펙트럼 계수가 디코딩 동안에 (예를 들어, 계수 예측 또는 계수 재정렬에 의해) 전처리되어 있는 경우, 이 계수들은 엔트로피 디코딩 이후에 (예를 들어, 계수 예측 또는 계수 재정렬에 의해) 후처리된다. 심볼들 각각은 양자화된 스펙트럼 계수를 나타낼 수 있다. 또는, 심볼들 각각은 일군의 양자화된 스펙트럼 계수를 나타낼 수 있다. 벡터 허프만 코딩의 경우, 심볼은, 예를 들어, 일군의 4개의 양자화된 스펙트럼 계수를 나타낸다. 런-레벨 디코딩의 경우, 심볼은, 예를 들어, 런-레벨 쌍을 나타낸다.

<380> 일련의 심볼에 대해, 디코더는 선택에 따라서는 선택된 엔트로피 모델을 알려주는 정보를 파싱한다(2710). 예를 들어, 전방 적응의 경우, 디코더는 인코더측 시그널링과 흡사한 메카니즘을 사용하여 선택된 엔트로피 모델을 알려주는 정보를 파싱한다.

<381> 디코더는 엔트로피 모델의 제1 세트로부터 엔트로피 모델을 선택한다(2720). 예를 들어, 디코더는 벡터 허프만 디코딩 또는 런-레벨 디코딩의 경우 다수의 이용가능한 허프만 코드 테이블 중에서 허프만 코드 테이블을 선택한다. 다른 대안으로서, 디코더는 다른 엔트로피 디코딩 방식에서 사용되는 엔트로피 모델을 선택한다. 어떤 구현에서, 디코더는 후방 적응에 대한 컨텍스트 정보에 따라 엔트로피 모델을 선택한다. 다른 구현에서, 디코더는 인코더에 의해 신호되고 비트스트림으로부터 파싱(2710)되는 정보에 기초하여 엔트로피 모델을 선택한다.

<382> 이어서, 디코더는 선택된 엔트로피 모델을 사용하여 일련의 심볼을 디코딩한다(2730). 엔트로피 모델에서의 임의의 전환점에서, 디코더는 하나 이상의 엔트로피 모델의 다른 세트로 전환할 수 있다. 예를 들어, 디코더는 제2 허프만 코드 테이블로의 전환을 나타내는 제1 허프만 코드 테이블에 대한 이스케이프 코드를 수신하고, 이어서 제2 허프만 코드 테이블을 사용하여 심볼을 디코딩한다.

<383> 인코더는 이어서 엔트로피 디코딩된 심볼에 대한 정보, 예를 들어, 차후의 처리를 위해 준비된 양자화된 스펙트럼 계수를 출력한다(2740).

<384> 디코더는 그 다음 계열을 계속할지를 판정하고(2750), '예'인 경우, 그 다음 계열의 심볼에 대한 엔트로피 모델을 선택한다(2710). 예를 들어, 한 구현에서 허프만 코드 테이블을 사용하여 양자화된 스펙트럼 계수를 디코딩할 때, 디코더는 바크 경계에서 코드 테이블을 변경할 수 있다. 디코딩 중인 현재 심볼에 대한 계수가 바크 경계를 넘어서는 경우(예를 들어, 심볼이 경계를 가로지르는 계수들의 벡터 또는 계수들의 런-레벨 쌍을 나타내기 때문에), 현재 심볼의 계수들의 끝이 유효한 변경 위치로 된다. 다른 대안으로서, 디코더는 다른 변경 위치에서 제1 모델 세트로부터 엔트로피 모델의 선택을 변경하고, 선택된 엔트로피 모델에 따라 디코딩된 일련의 심볼이 어떤 다른 기간을 갖는다.

<385> **E. 결과**

<386> 확률이 낮은 심볼값에 대해 근사화된 분포를 사용하여 코딩하는 것은 인코더 및 디코더에서 분포 또는 코드 테이블을 위해 필요한 메모리를 절감할 수 있게 해준다. 섹션 V.A.1의 분석 프레임워크의 관점에서, 인코더 및 디코더는 $P_{S(j),X(q)}$에 대한 분포 및/또는 코드 테이블을 저장한다. 즉, 인코더 및 디코더는 집합 Q 내의 심볼값 $X(i)$에 대한 상태 $S(j)$마다의 분포 및/또는 테이블을 저장한다. 집합 R 내의 심볼값 $X(i)$에 대해, 인코더 및 디코더는 하나의 분포 $P'_{S(j),X(i),R}$에 대한 분포 및/또는 테이블을 저장한다.

<387> 테이블이 각각의 상태에 대해 B 바이트의 메모리를 차지하고 16개의 상태가 있는 것으로 가정하자. 그러면, 일반적인 채워진 테이블 경우에, 인코더 및 디코더는 각각 16개 테이블에 대해 16*B 바이트의 메모리를 필요로 한다. 그렇지만, (집합 Q 내에서) 심볼값의 10%만이 확률이 높은 것으로 지정되어 있는 경우, 필요한 메모리의 간단한 근사값은 (16*B*.1)+(B*.9)=2.5*B이다. 따라서, 필요한 메모리는 6배 이상 감소되었으며, 엔트로피 코딩 이득은 채워진 테이블 경우와 비교하여 약간만 감소되었다.

산업상 이용 가능성

<388> 개시된 발명의 원리들이 적용될 수 있는 많은 가능한 실시예들을 살펴보면, 예시된 실시예들이 본 발명의 양호한 예에 불과하고 본 발명의 범위를 제한하는 것으로 해석되어서는 안된다는 것을 잘 알 것이다. 오히려, 본 발명의 범위는 이하의 청구항들에 의해 정의된다. 따라서, 우리의 발명은 모두가 이들 청구항의 범위 및 정신 내에 속하는 것으로 보아야 한다.

도면의 간단한 설명

<36> 도 1은 다양한 기술된 실시예들이 구현될 수 있는 일반화된 운영 환경의 블록도.

<37> 도 2 내지 도 7은 다양한 기술된 실시예들이 구현될 수 있는 일반화된 인코더 및/또는 디코더의 블록도.

<38> 도 8a 및 도 8b는 각각 다중-채널 오디오 신호 및 대응하는 윈도우 구성을 나타낸 차트.

<39> 도 9 및 도 10은 각각 시간 노이즈 쉐이핑을 갖는 인코더 및 디코더를 나타낸 블록도.

<40> 도 11 및 도 12는 각각 비트 레이트 감소를 위한 계수 예측을 갖는 인코더 및 디코더를 나타낸 블록도.

<41> 도 13 및 도 14는 각각 양자화된 스펙트럼 계수의 코딩 및 디코딩에서의 계수 예측 기법을 나타낸 플로우차트.

<42> 도 15a 및 도 15b는 각각 시간 영역에서의 주기적인 오디오 신호 및 대응하는 스펙트럼 계수를 나타낸 차트.

<43> 도 16 및 도 17은 각각 계수 재정렬(coefficient reordering)을 갖는 인코더 및 디코더를 나타낸 블록도.

<44> 도 18a 내지 도 18c는 엔트로피 인코딩 이전에 스펙트럼 계수를 재정렬하는 기법을 나타낸 플로우차트.

<45> 도 19a 내지 도 19c는 엔트로피 디코딩 후에 스펙트럼 계수를 재정렬하는 기법을 나타낸 플로우차트.

<46> 도 20은 재정렬 이후에 도 15b의 스펙트럼 계수를 나타낸 차트.

<47> 도 21은 예시적인 오디오 파일의 서브-프레임별 계수 재정렬로 인한 코딩 이득을 나타낸 차트.

<48> 도 22는 계층적으로 구성된 엔트로피 모델을 나타낸 도면.

<49> 도 23은 심볼값의 근사 분포에 대한 허프만 코드를 나타낸 차트.

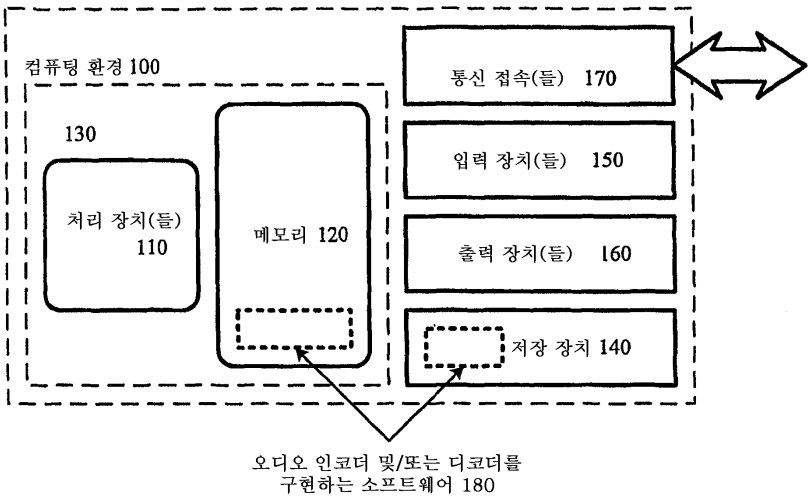
<50> 도 24 및 도 25는 확률 분포에 대한 훈련 벡터를 클러스터링하는 기법을 나타낸 플로우차트.

<51> 도 26은 다수의 엔트로피 모델의 선택적 사용으로 인코딩하는 기법을 나타낸 플로우차트.

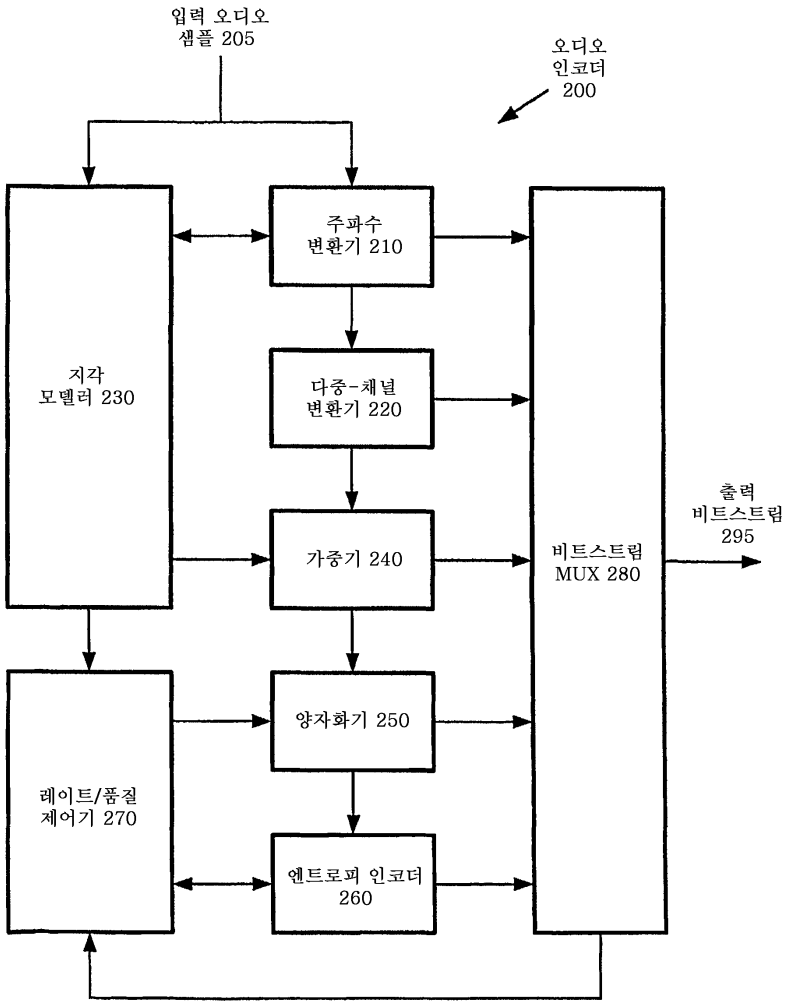
<52> 도 27은 다수의 엔트로피 모델의 선택적 사용으로 디코딩하는 기법을 나타낸 플로우차트.

도면

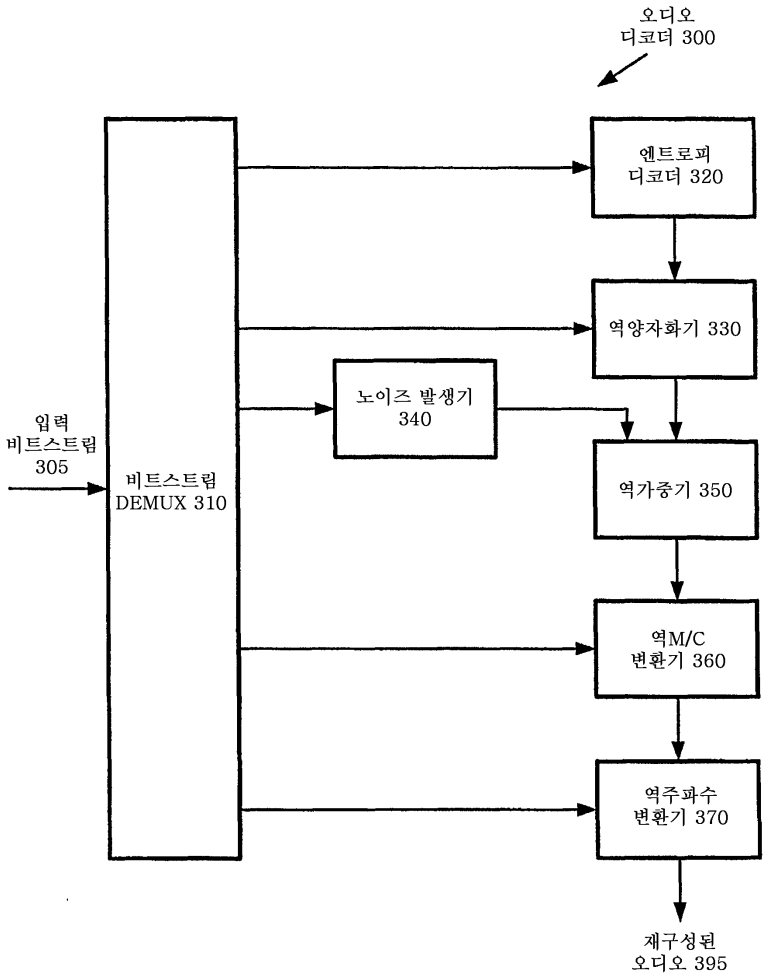
도면1



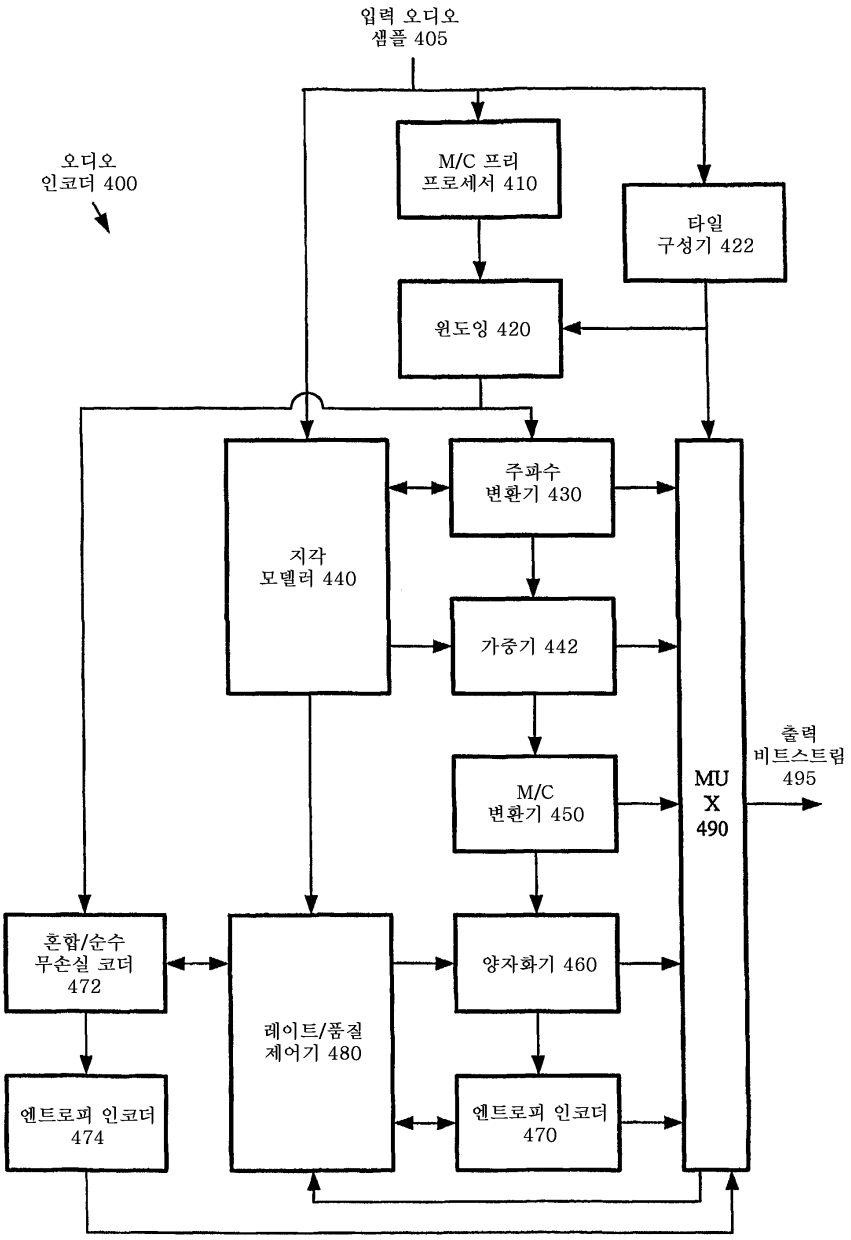
도면2


입력 오디오
샘플 205
오디오
인코더
200
주파수
변환기 210
다중-채널
변환기 220
가중기 240
지각
모델러 230
양자화기 250
엔트로피 인코더
260
레이트/품질
제어기 270
비트스트림
MUX 280
출력
비트스트림
295

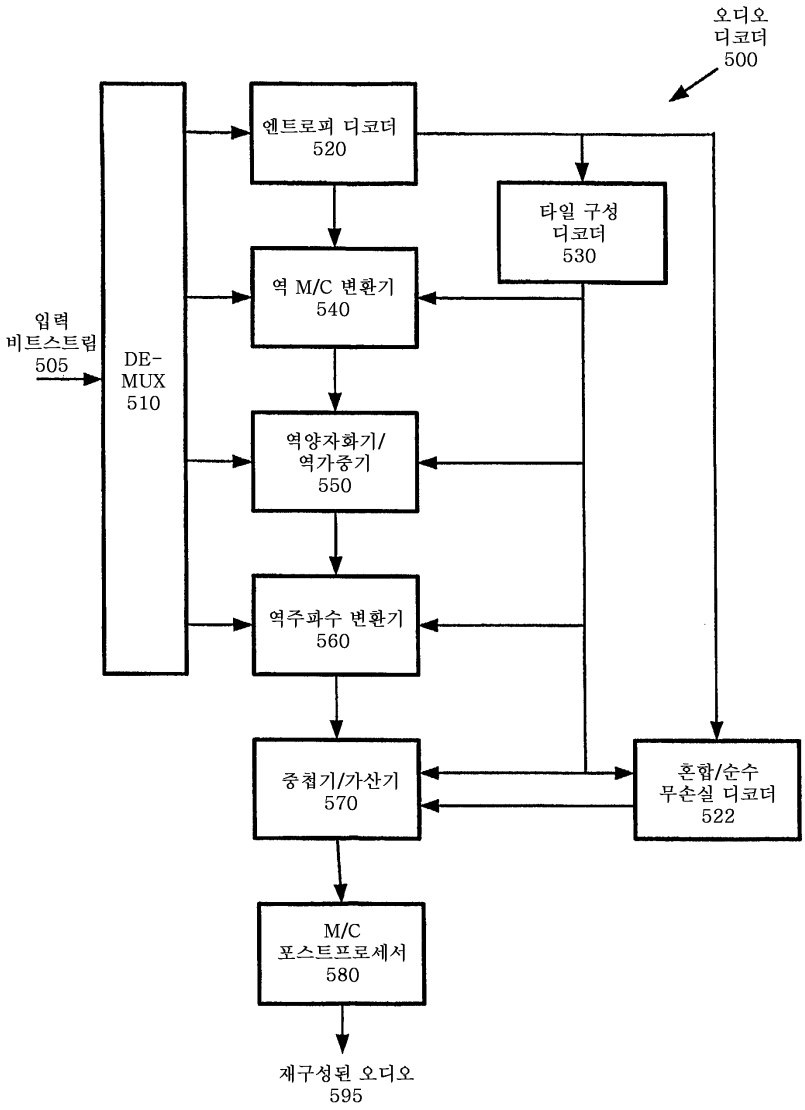
도면3


오디오
디코더 300
엔트로피
디코더 320
역양자화기 330
노이즈 발생기
340
역가중기 350
역M/C
변환기 360
역주파수
변환기 370
재구성된
오디오 395
입력
비트스트림
305
비트스트림
DEMUX 310

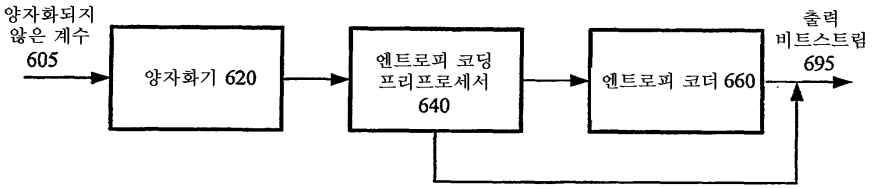
도면4


입력 오디오
샘플 405
오디오
인코더 400
M/C 프리
프로세서 410
타일
구성기 422
윈도잉 420
주파수
변환기 430
지각
모델러 440
가중기 442
M/C
변환기 450
MU
X
490
출력
비트스트림
495
혼합/순수
무손실 코더
472
레이트/품질
제어기 480
양자화기 460
엔트로피 인코더
474
엔트로피 인코더
470

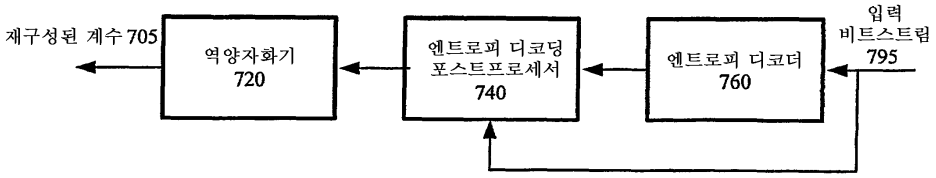
도면5


오디오
디코더
500
엔트로피 디코더
520
타일 구성
디코더
530
역 M/C 변환기
540
입력
비트스트림
505
DE-
MUX
510
역양자화기/
역가중기
550
역주파수 변환기
560
중첩기/가산기
570
혼합/순수
무손실 디코더
522
M/C
포스트프로세서
580
재구성된 오디오
595

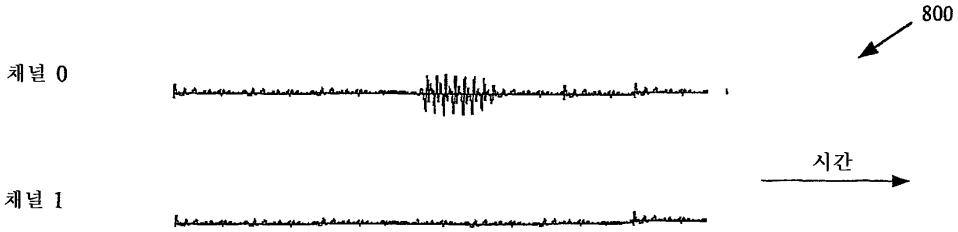
도면6


양자화되지
않은 계수
605
양자화기 620
엔트로피 코딩
프리프로세서
640
엔트로피 코더 660
출력
비트스트림
695

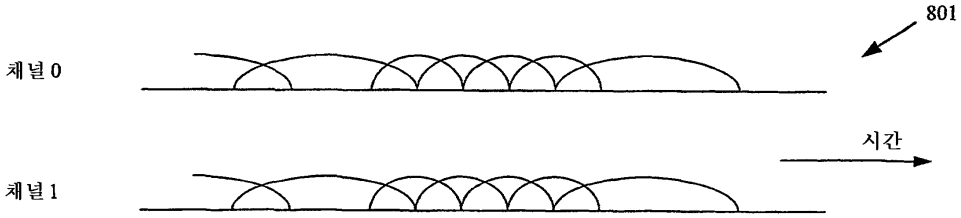
도면7



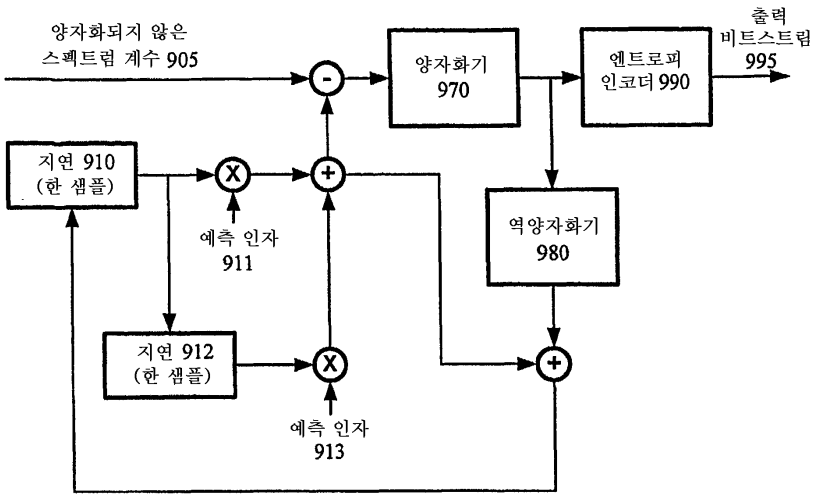
도면8a



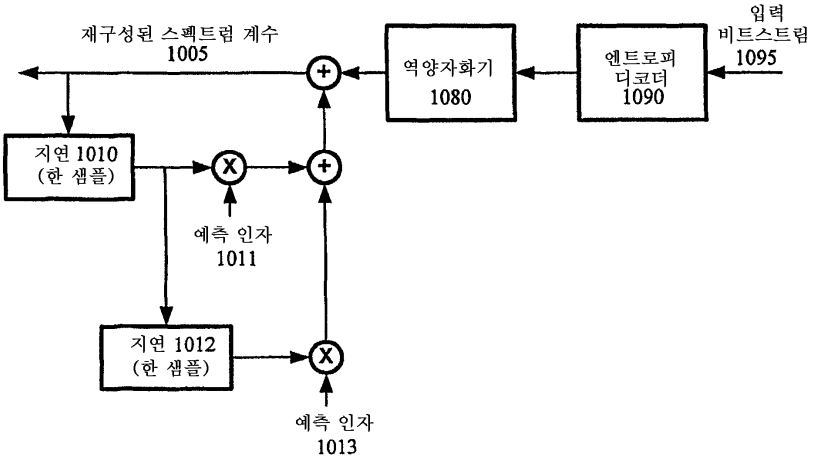
도면8b



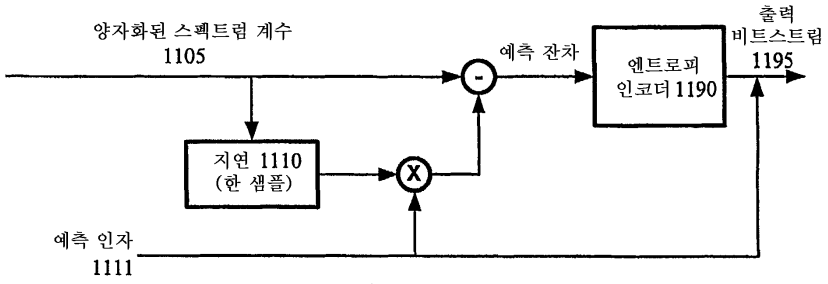
도면9



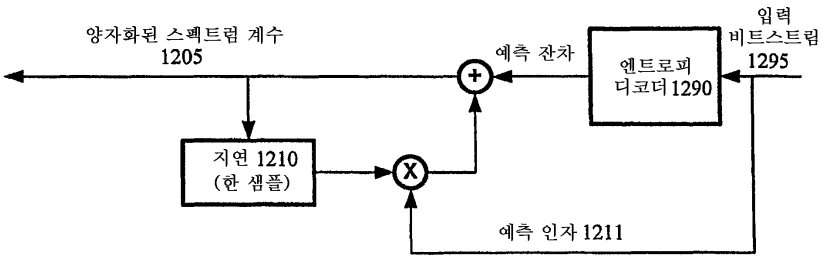
도면10



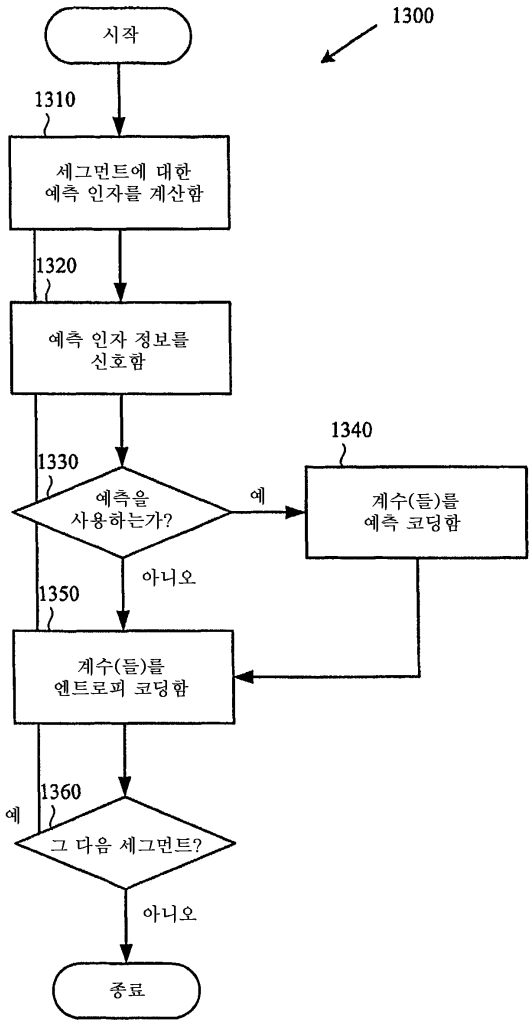
도면11



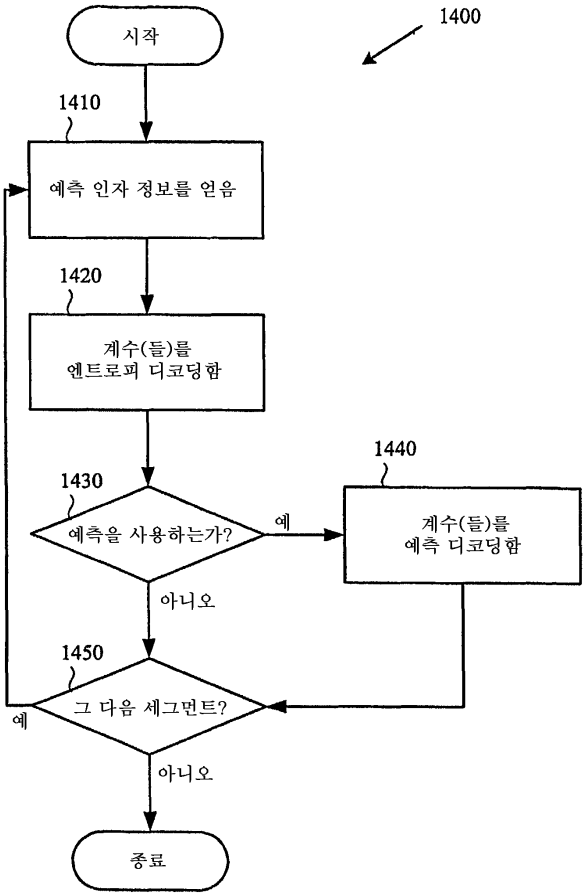
도면12



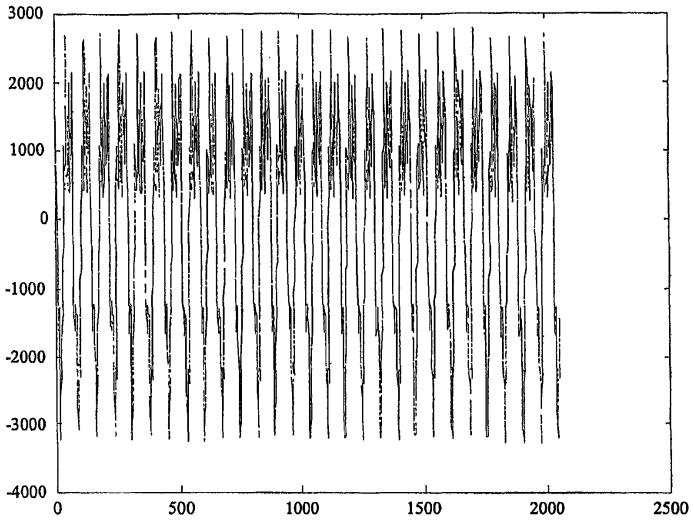
도면13


1300
시작
1310
세그먼트에 대한
예측 인자를 계산함
1320
예측 인자 정보를
신호함
1330
예측을
사용하는가?
예
1340
계수(들)를
예측 코딩함
아니오
1350
계수(들)를
엔트로피 코딩함
1360
예
그 다음 세그먼트?
아니오
종료

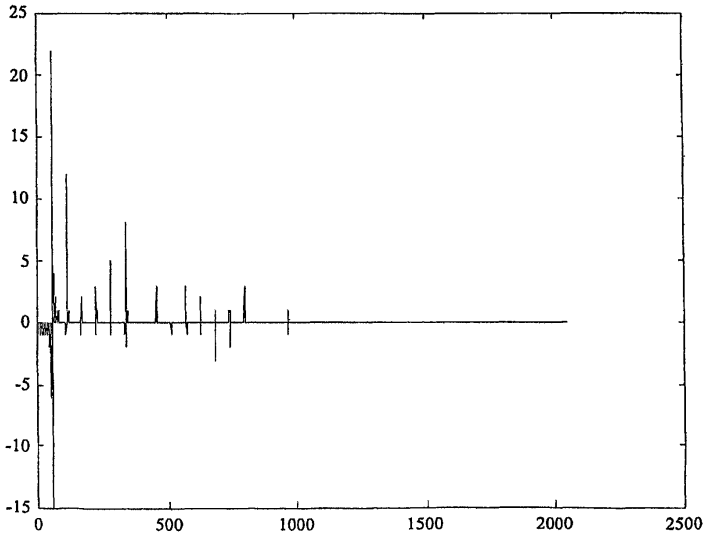
도면14


1400
시작
1410
예측 인자 정보를 얻음
1420
계수(들)를
엔트로피 디코딩함
1430
예측을 사용하는가?
예
1440
계수(들)를
예측 디코딩함
아니오
1450
그 다음 세그먼트?
예
아니오
종료

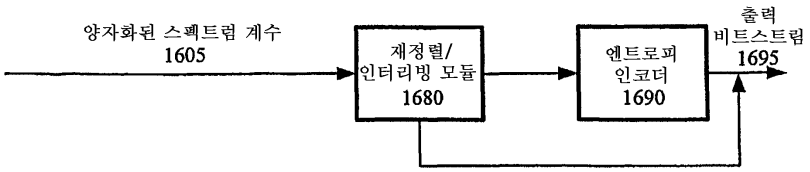
도면15a


3000
2000
1000
0
-1000
-2000
-3000
-4000
0
500
1000
1500
2000
2500

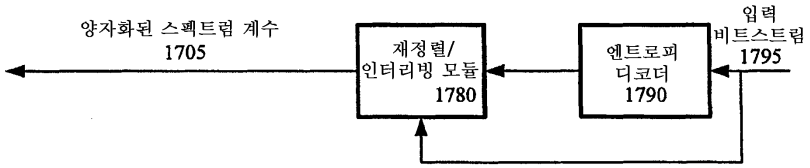
도면15b



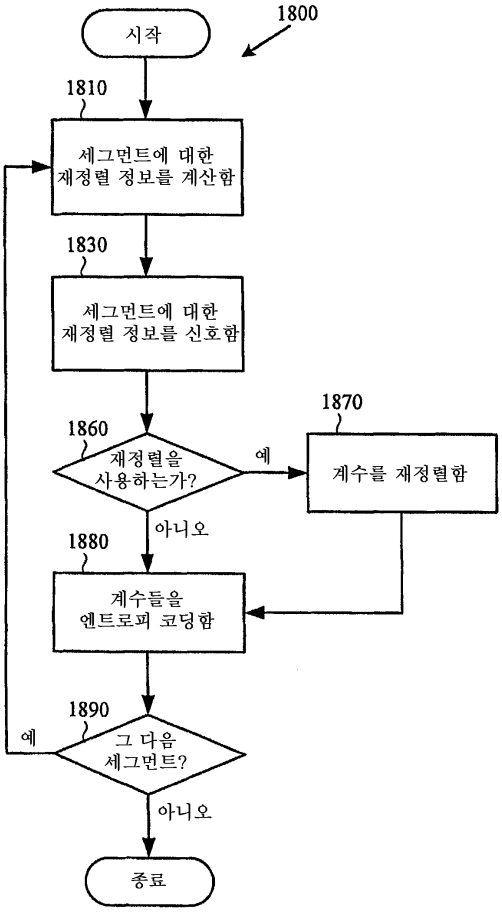
도면16



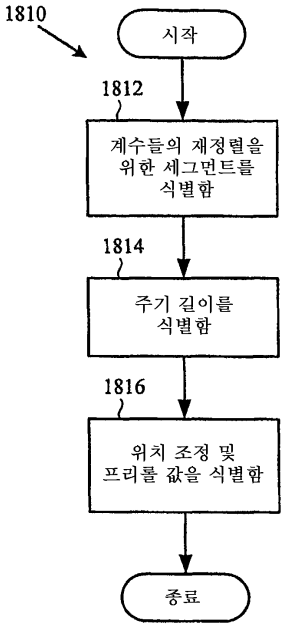
도면17



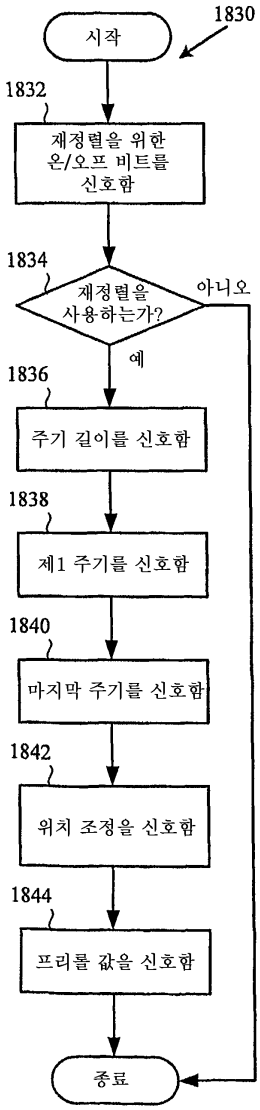
도면18a


1800
시작
1810
세그먼트에 대한 재정렬 정보를 계산함
1830
세그먼트에 대한 재정렬 정보를 신호함
1860
재정렬을 사용하는가?
예
1870
계수를 재정렬함
아니오
1880
계수들을 엔트로피 코딩함
1890
그 다음 세그먼트?
예
아니오
종료

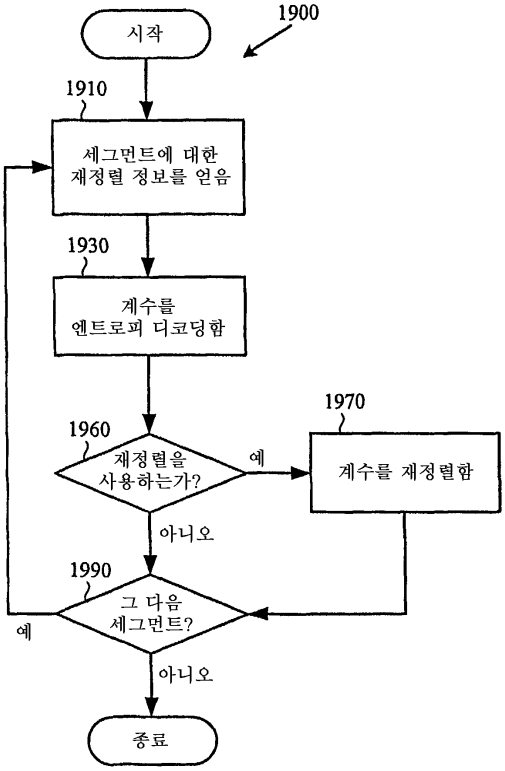
도면18b


1810
시작
1812
계수들의 재정렬을 위한 세그먼트를 식별함
1814
주기 길이를 식별함
1816
위치 조정 및 프리롤 값을 식별함
종료

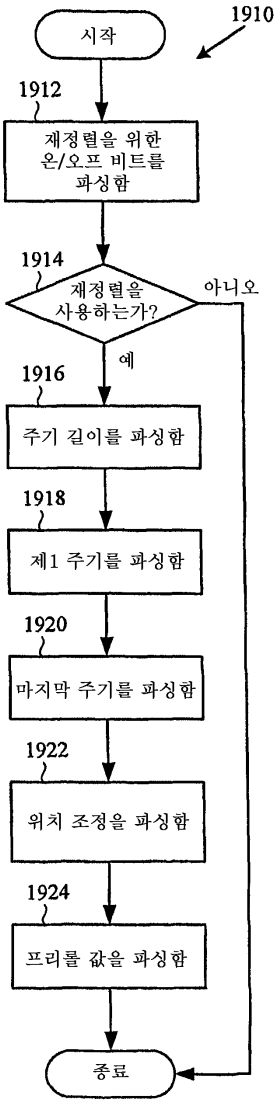
도면18c


1830
시작
1832
재정렬을 위한
온/오프 비트를
신호함
1834
재정렬을
사용하는가?
아니오
예
1836
주기 길이를 신호함
1838
제1 주기를 신호함
1840
마지막 주기를 신호함
1842
위치 조정을 신호함
1844
프리를 값을 신호함
종료

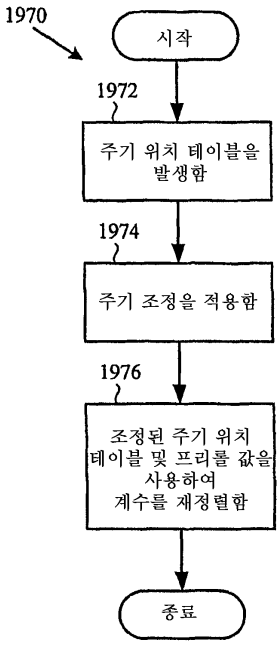
도면19a


1900
시작
1910
세그먼트에 대한
재정렬 정보를 얻음
1930
계수를
엔트로피 디코딩함
1960
재정렬을
사용하는가?
예
1970
계수를 재정렬함
아니오
1990
그 다음
세그먼트?
예
아니오
종료

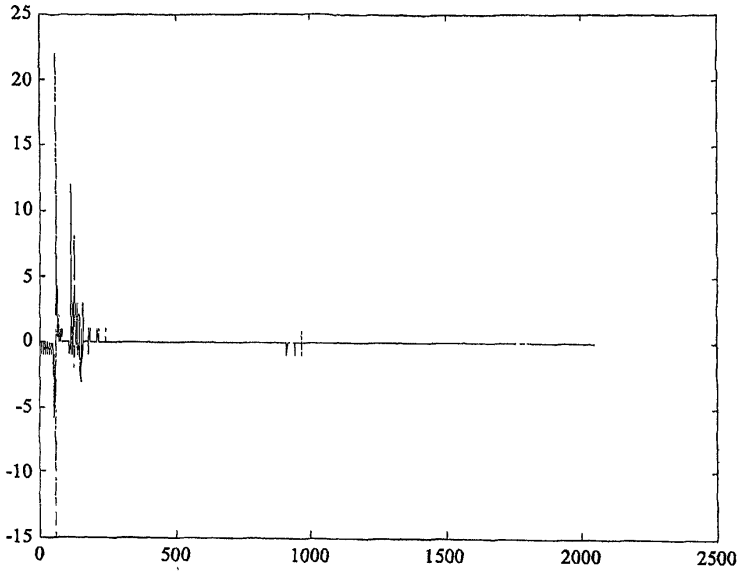
도면19b


1910
시작
1912
재정렬을 위한 온/오프 비트를 파싱함
1914
재정렬을 사용하는가?
아니오
예
1916
주기 길이를 파싱함
1918
제1 주기를 파싱함
1920
마지막 주기를 파싱함
1922
위치 조정을 파싱함
1924
프리롤 값을 파싱함
종료

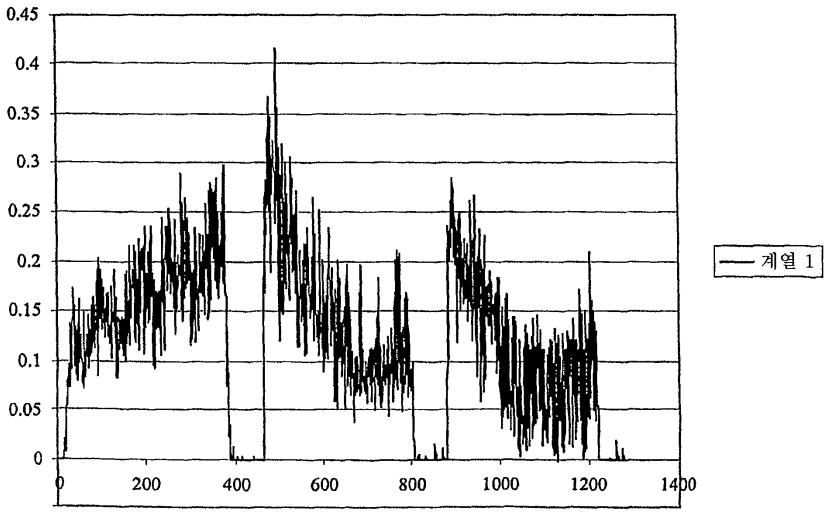
도면19c


1970
시작
1972
주기 위치 테이블을 발생함
1974
주기 조정을 적용함
1976
조정된 주기 위치 테이블 및 프리롤 값을 사용하여 계수를 재정렬함
종료

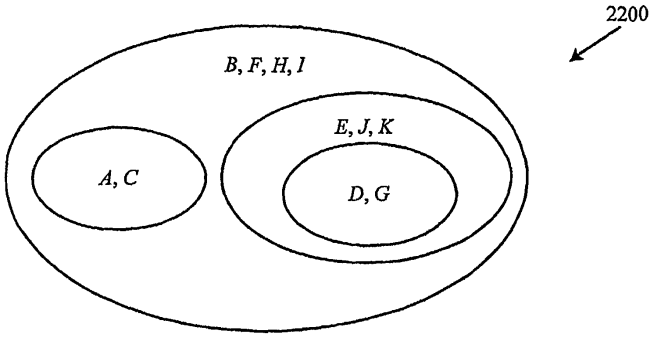
도면20


25
20
15
10
5
0
-5
-10
-15
0
500
1000
1500
2000
2500

도면21

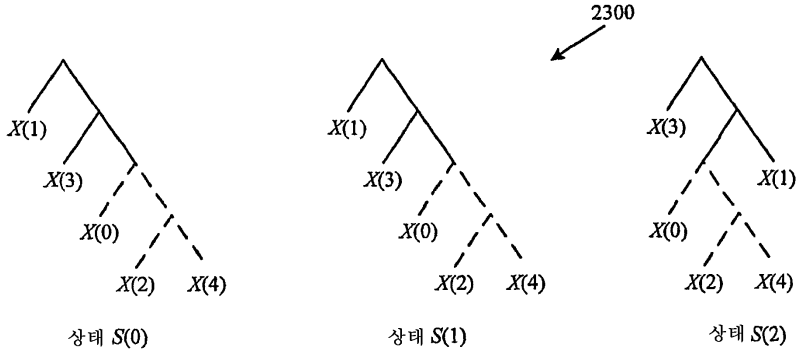


도면22

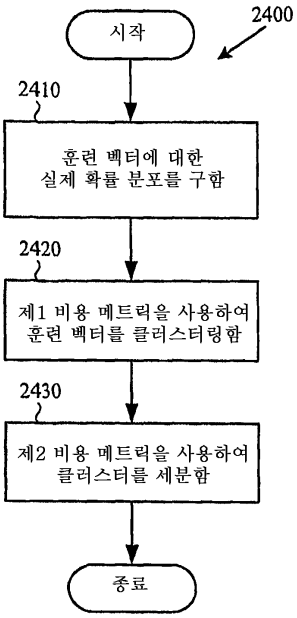


제1 모델 세트: {B, F, H, I, 전환_1, 전환_2}에 대한 8개의 엔트로피 모델
제2 모델 세트: 전환_1 이후의 {A, C}에 대한 3개의 엔트로피 모델
제3 모델 세트: 전환_2 이후의 {E, J, K, 전환_3}에 대한 4개의 엔트로피 모델
제4 모델 세트: 전환_3 이후의 {D, G}에 대한 1개의 엔트로피 모델

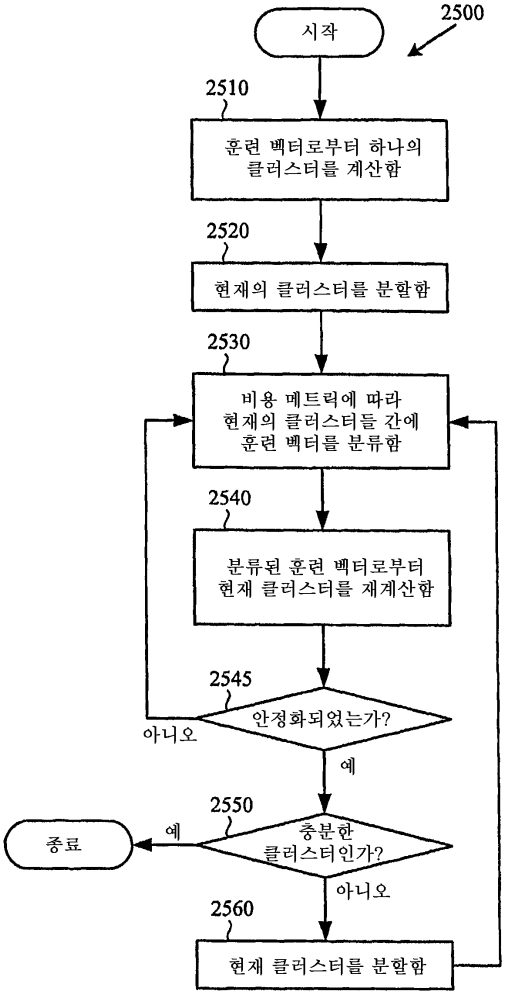
도면23



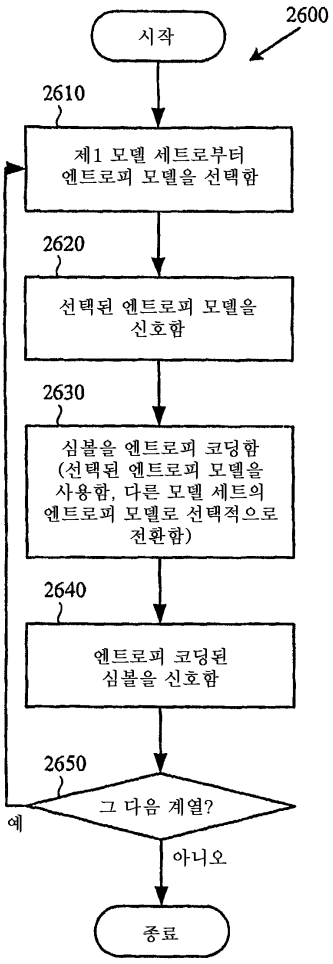
도면24


시작
2400
2410
훈련 벡터에 대한
실제 확률 분포를 구함
2420
제1 비용 메트릭을 사용하여
훈련 벡터를 클러스터링함
2430
제2 비용 메트릭을 사용하여
클러스터를 세분함
종료

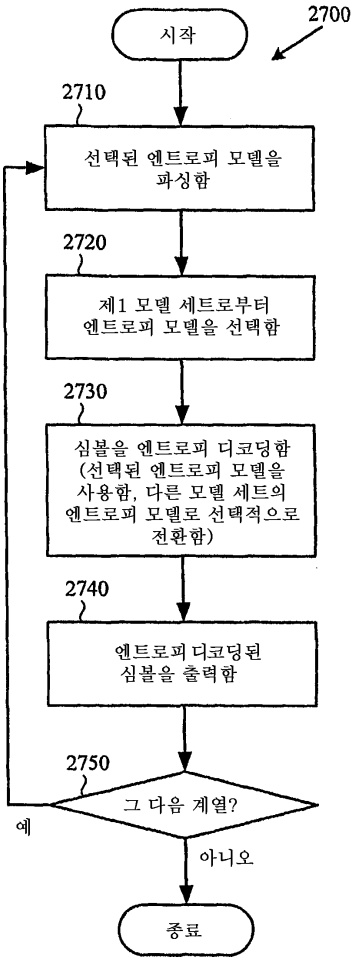
도면25


2500
시작
2510
훈련 벡터로부터 하나의 클러스터를 계산함
2520
현재의 클러스터를 분할함
2530
비용 메트릭에 따라 현재의 클러스터들 간에 훈련 벡터를 분류함
2540
분류된 훈련 벡터로부터 현재 클러스터를 재계산함
2545
안정화되었는가?
아니오
예
2550
충분한 클러스터인가?
예
종료
아니오
2560
현재 클러스터를 분할함

도면26


2600
시작
2610
제1 모델 세트로부터
엔트로피 모델을 선택함
2620
선택된 엔트로피 모델을
신호함
2630
심볼을 엔트로피 코딩함
(선택된 엔트로피 모델을
사용함, 다른 모델 세트의
엔트로피 모델로 선택적으로
전환함)
2640
엔트로피 코딩된
심볼을 신호함
2650
그 다음 계열?
예
아니오
종료

도면27


2700
시작
2710
선택된 엔트로피 모델을 파싱함
2720
제1 모델 세트로부터 엔트로피 모델을 선택함
2730
심볼을 엔트로피 디코딩함 (선택된 엔트로피 모델을 사용함, 다른 모델 세트의 엔트로피 모델로 선택적으로 전환함)
2740
엔트로피 디코딩된 심볼을 출력함
2750
그 다음 계열?
예
아니오
종료