



República Federativa do Brasil
Ministério da Economia
Instituto Nacional da Propriedade Industrial

(21) PI 0910796-7 A2



(22) Data do Depósito: 18/05/2009

(43) Data da Publicação Nacional: 03/11/2020

(54) Título: CODIFICADOR DE ÁUDIO E DECODIFICADOR DE ÁUDIO

(51) Int. Cl.: G10L 19/02.

(30) Prioridade Unionista: 08/10/2008 US 61/103,820; 11/07/2008 US 61/079,842.

(71) Depositante(es): FRAUNHOFER-GESELLSCHAFT ZUR FÖRDERUNG DER ANGEWANDTEN FORSCHUNG E.V..

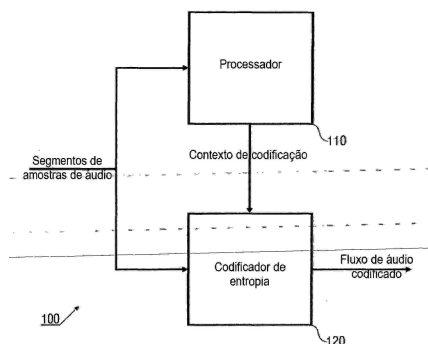
(72) Inventor(es): MULTRUS, MARKUS.; GRILL, BERNHARD.; FUCHS, GUILLAUME.; GEYERSBERGER, STEFAN.; RETTELBACH, NIKOLAUS.; BACIGALUPO, VIRGILIO..

(86) Pedido PCT: PCT EP2009003521 de 18/05/2009

(87) Publicação PCT: WO 2010/003479 de 14/01/2010

(85) Data da Fase Nacional: 10/01/2011

(57) Resumo: CODIFICADOR DE ÁUDIO E DECODIFICADOR DE ÁUDIO. Um decodificador de áudio (110) para segmentos de codificação de coeficientes, os segmentos de coeficientes representando resoluções diferentes de tempo e frequência de um sinal de áudio amostrado, o codificador de áudio (100) compreendendo um processador (110) para derivar um contexto de codificação para o coeficiente codificado atual baseado em um coeficiente codificado anterior de um segmento anterior, o coeficiente previamente codificado representando uma diferente resolução de tempo ou frequência do que o coeficiente codificado atual. O codificador de áudio (100) compreende ainda um codificação de entropia (120) para codificação de entropia do atual coeficiente baseado no contexto de codificação para obter um fluxo de áudio codificado.



"CODIFICADOR DE ÁUDIO E DECODIFICADOR DE ÁUDIO"

ESPECIFICAÇÃO

A presente invenção está no campo de codificação de áudio, especialmente no campo de codificação de entropia baseada em contexto.

Os conceitos tradicionais de codificação de áudio incluem um esquema de codificação de entropia para redução de redundância. Tipicamente, a codificação de entropia é aplicada a coeficientes espectrais quantizados para esquemas de codificação baseados em domínio de frequência ou amostras de domínio de tempo quantizadas para esquemas de codificação baseados em domínio de tempo. Esses esquemas de codificação de entropia tipicamente fazem uso da transmissão de uma palavra-código em combinação com um índice de codebook de acordo, que permite a um decodificador procurar uma certa página do codebook para decodificar uma palavra de informação codificada correspondente a palavra-código transmitida em tal página. Em alguns conceitos de codificação, a transmissão do índice do codebook pode não ser obrigatória, por exemplo, para casos em que o índice do codebook pode ser determinado a partir do contexto e um símbolo, que é, por exemplo, codificado por entropia como descrito em Meine, Edler, "Improved Quantization and Lossless Coding for Subband Audio Coding" e Meine, "Vektorquantisierung und kontextabhängige arithmetische Codierung für MPEG-4 AAC", Dissertação, Gottfried Wilhelm Leibnitz Universität Hannover, Hanover 2007.

Para codificação de áudio baseada em domínio espectral ou de frequência, um contexto pode descrever símbolos ou propriedades estatísticas, como por exemplo, coeficientes

espectrais quantizados, que são anteriores em tempo e/ou frequência. Em alguns dos conceitos convencionais, esses símbolos podem estar disponíveis em ambos, o lado do codificador e do decodificador, e baseado nesses símbolos, um codebook ou contexto
 5 pode ser determinado de forma síncrona no lado do decodificador e do codificador.

A Fig. 9 deve ilustrar um exemplo de um contexto e suas dependências. A Fig. 9 mostra um plano de frequência de tempo, no qual um número de símbolos é indicado. Símbolo $S_{n,m}$ denota
 * 10 um símbolo no tempo n e frequência m . A Fig. 9 ilustra que para codificar um certo símbolo, seu contexto é usado para determinar o codebook associado. Por exemplo, para um símbolo S_{n_0,m_0} , seriam
 todos os símbolos com

$n < n_0$ e qualquer m , ou com $n = n_0$ e $m < m_0$.

15 Em implementações práticas, um contexto não é infinito, mas limitado. No exemplo descrito na Fig. 9, o contexto para o símbolo $S_{0,3}$ poderia ser, por exemplo,

$S_{0,2}, S_{0,1}, S_{-1,5}, S_{-1,4}, S_{-1,3}, S_{-1,2}, S_{-1,1}, S_{-2,5}, S_{-2,4}, S_{-2,3}, S_{-2,2}, S_{-2,1}$.

20 Para codificação de áudio baseada em frequência, variante de tempo, bancos de filtros adaptativos de sinal ou os chamados de transformações de bloco podem ser usados, como por exemplo, descrito em Edler, B., "Codierung von Audiosignalen mit überlappender Transformation und adaptiven Fensterfunktionen",
 25 Frequenz, Ausgabe 43, Setembro 1989.

Em outras palavras, mudanças de resolução de frequência/tempo podem ocorrer com o tempo dentro desses conceitos de codificação de áudio. Um conceito popular de codificação de

áudio é o chamado de AAC (AAC = Codificação de Áudio Avançada), nesse caso dois comprimentos de bloco são usados, para quais são codificados por exemplo tanto coeficientes de transformada de 128 ou 1024 representando os componentes de frequência de amostras de domínio de tempo janeladas de 256 ou 2048, respectivamente.

Esses conceitos permitem uma comutação entre as resoluções diferentes, dependendo de certas características do sinal, como por exemplo ocorrência de transitórios ou tonalidade ou se o sinal lembra música ou fala, etc. No caso de comutação entre resoluções diferentes de tempo/frequência, como por exemplo entre tipos de diferentes de bloco AAC, o contexto não é consistente. Os conceitos convencionais ou implementações no estado-da-arte podem utilizar redefinição de contexto, ou seja, é basicamente alternado para um estado, no qual nenhum contexto está disponível, no qual o contexto é construído desde o começo. Essa abordagem pode funcionar de forma suficientemente bem, por exemplo em AAC, já que garante ao menos dois longos blocos ou oito blocos curtos em sequência, onde pode presumir-se que comutação raramente ocorre.

Contudo, conceitos convencionais redefinindo o contexto são em geral sub-adequados em termos da eficiência da codificação, já que cada vez que o contexto é redefinido, a seleção do codebook subsequente é baseado nos valores, que são planejados como soluções de recuo para contexto desconhecido. Geralmente, codebooks sub-adequados são, então, selecionados. O recuo na eficiência de codificação pode ser ínfima para casos que comutação ocorre raramente. Para um cenário, contudo, com mais comutação frequente, isso leva a uma perda significativa da

eficiência de codificação. De um lado, uma comutação mais frequente é fortemente desejada para taxas mais baixas de dados/taxas de amostragem, já que especialmente aqui, uma adaptação adequada do comprimento da transformada para a
5 característica do sinal é desejada. Do outro lado, a eficiência de codificação diminui significativamente quando a comutação ocorre frequentemente.

É objeto da presente invenção fornecer um conceito para comutação entre comprimentos diferentes de
10 transformada na codificação de áudio fornecendo uma eficiência aprimorada de codificação.

O objeto é alcançado por um codificador de áudio
de acordo com a reivindicação 1, um método para codificação de
áudio de acordo com a reivindicação 8, um decodificador de áudio
15 para reivindicação 9, e um método para decodificação de áudio de
acordo com reivindicação 16.

A presente invenção é baseada na descoberta que em codificação baseada em contexto como, por exemplo, codificação de entropia baseada em contexto, que pode ser aplicada à
20 resoluções diferentes de tempo/frequência, um mecanismo de mapeamento de contexto pode ser usado, no caso de mudança com o tempo da resolução de tempo/frequência, com isso alcançando uma eficiência melhorada de codificação. É uma descoberta da presente invenção que quando a comutação entre resoluções diferentes de
25 frequência ou tempo, contextos para coeficientes tendo que as novas resoluções podem ser derivadas de coeficientes tendo as velhas resoluções. É uma descoberta da presente invenção que, por exemplo, interpolação, extrapolação, sub-amostragem, downsampling,

upsampling, etc, podem ser usados para adaptação de contexto e/ou derivação quando acontece a comutação de resoluções de tempo/frequência na codificação de áudio.

As materializações da presente invenção fornecem um método de mapeamento, que mapeia a frequência ou coeficientes espectrais de um contexto armazenado, que se refere a uma resolução velha, para a resolução de frequência de um contexto atual ou uma estrutura atual. Em outras palavras, as informações de contexto anterior podem ser utilizadas para determinação do codebook, ou seja, para obter informações sobre novo contexto. Materializações podem com isso permitir uma troca mais frequente do comprimento do bloco e, portanto, uma melhor adaptação às características do sinal sem perder a eficiência de codificação.

Manifestações da presente invenção serão detalhadas com as figuras de acompanhamento, em que

Fig. 1 Mostra um materialização de um codificador de áudio;

Fig. 2 Mostra um materialização de um decodificador de áudio;

Fig. 3 Mostra uma materialização para um upsampling de contexto

Fig. 4 Mostra uma materialização para um downsampling de contexto;

Fig. 5 Ilustra resoluções de tempo e frequência da comutação de áudio;

Fig. 6 Ilustra uma implementação de uma materialização;

Fig. 7a Mostra um fluxograma de uma materialização de um método para codificação;

Fig. 7b Ilustra o procedimento de atualização do contexto geral de uma materialização;

5 Fig. 7c Ilustra o procedimento de atualização do contexto de uma materialização para mudanças de resolução;

Fig. 8 Mostra um fluxograma de uma materialização de um método para codificação; e

Fig. 9 Mostra um esquema de codificação de
10 tempo frequência do estado da arte.

A Fig. 1 mostra uma materialização de um codificador de áudio 100 para os segmentos de codificação de coeficientes, os segmentos de coeficientes representando diferentes resoluções de tempo ou frequência de um sinal de áudio

15 amostrado. O codificador de áudio 100 dispõe de um processador 110 para obter um contexto de codificação para um coeficiente

codificado de um segmento atual com base em um coeficiente previamente codificado de um segmento anterior, o coeficiente previamente codificado representando uma diferente resolução de
20 tempo ou frequência do que o coeficiente codificado atual. A materialização do codificador de áudio inclui ainda um codificador de entropia 120 para codificação de entropia do atual coeficiente baseado no contexto de codificação para obter um fluxo de áudio codificado.

25 Em materializações, os coeficientes podem corresponder às amostras de áudio, amostras quantizadas de áudio, coeficientes espectrais ou de frequência, coeficientes escalonados, coeficientes filtrados ou transformados etc. ou

qualquer combinação destes.

Em materializações, o codificador de áudio 100 pode ainda compreender um meio para fornecer os segmentos dos coeficientes de um fluxo de áudio, os coeficientes formando uma
5 representação espectral de um sinal de áudio em uma resolução espectral variando entre os coeficientes. Os meios de fornecer os segmentos podem ser adaptados para determinar os segmentos baseados em comprimentos de janelas diferentes de domínio de tempo ou estruturas diferentes de áudio, ou seja, estruturas de áudio
10 tendo diferentes tamanhos ou números diferentes de coeficientes por largura de banda, ou seja, uma resolução espectral ou de frequência diferente. Os meios para fornecer podem ser adaptados para determinar segmentos de tempo 1024 e 128, coeficientes espectrais, de frequência e tempo.

15 O processador 110 pode, em materializações, ser adaptado para derivar o contexto de codificação baseado nas representações do domínio espectral ou de frequência dos coeficientes e segmentos atuais e anteriores. Em outras palavras, em materializações os segmentos sucessivos podem ser representados
20 em tempo e/ou frequência diferente ou domínios espectrais. O processador 110 pode ser adaptado para derivar o contexto de codificação por frequência ou faixa espectral do segmento atual, por exemplo baseado em coeficientes espectrais vizinhos de segmentos anteriores e/ou segmento atual. Em materializações, os
25 segmentos podem ser determinados inicialmente no domínio de tempo, por exemplo, através do janelamento do fluxo de áudio. Baseado nesses segmentos ou coeficientes de domínio de tempo segmentos ou coeficientes de domínio espectral pode ser determinado por meio de

transformação. Os segmentos podem ser representados no domínio espectral ou de frequência nos termos de energia, uma amplitude e fase, uma amplitude e sinal, etc. Por frequência ou faixa espectral, ou seja, os segmentos podem ser subdivididos em 5 diferentes faixas espectrais ou de frequência. O processador 110 pode então derivar contextos de código por frequência ou faixa espectral em algumas materializações.

O processador 110 e o codificador de entropia 120 podem ser configurados para operar baseados em um downsampling de 10 frequência ou coeficientes espectrais de um segmento anterior quando o segmento anterior pertencendo ao contexto compreende uma melhor resolução espectral ou de frequência do que o segmento atual. Em materializações, o processador 110 e o codificador de entropia 120 podem ser configurados para operar baseados em um 15 upsampling de frequência ou coeficientes espectrais de um segmento anterior quando o segmento anterior pertencendo ao contexto compreende uma melhor resolução espectral ou de frequência do que o segmento atual.

Materializações podem fornecer um método para 20 codificação de segmentos de coeficientes representando diferentes resoluções de tempo ou frequência de um sinal de áudio amostrado. O método pode compreender uma etapa de derivação de um contexto de codificação para um coeficiente atual ou atualmente codificado de um segmento atual baseado em um coeficiente anterior ou 25 anteriormente codificado de um segmento anterior e opcionalmente também está baseado no coeficiente atual ou atualmente codificado, onde o coeficiente anterior ou anteriormente codificado representa uma resolução diferente do que o coeficiente atual ou atualmente

codificado. O método pode ainda compreender uma etapa da codificação de entropia de um coeficiente atual baseado no contexto de codificação para obter um fluxo de áudio codificado.

De forma correspondente, as materializações podem
5 compreender um decodificador de áudio 200 do qual uma materialização é descrita na Fig. 2. O decodificador de áudio 200 é adaptado para decodificação de um fluxo de áudio codificado para obter segmentos de coeficientes representando resoluções diferentes de tempo ou frequência de um sinal de áudio amostrado,
10 o decodificador de áudio 200 compreende um processador 210 para derivar um contexto de codificação para um coeficiente atual ou atualmente decodificado baseado em um coeficiente atual ou ~~atualmente decodificado, o coeficiente anterior ou anteriormente~~ decodificado representando uma resolução diferente de tempo e
15 frequência do que o coeficiente atualmente decodificado. Além do mais, o decodificador de áudio 200 compreende de um de codificador de áudio 220 para decodificação de entropia do atual coeficiente baseado no contexto de codificação e no fluxo de áudio codificado.

Em materializações, o decodificador de áudio 200
20 compreende um decodificador de entropia 220, que é adaptado para determinar os segmentos dos coeficientes decodificados baseados em comprimentos diferentes de janela de domínio de tempo ou comprimentos diferentes de estrutura de áudio. O decodificador de entropia 220 pode ser adaptado para determinar segmentos de, por
25 exemplo, 1024 e 128, amostras de domínio de tempo ou frequência ou coeficientes espectrais. De forma correspondente, o processador 210 pode ser adaptado para derivar o contexto de codificação baseado em uma representação de domínio espectral ou de frequência

de coeficientes de segmentos anteriores e/ou o segmento atual.

Em materializações, o processador 210 pode ser adaptado para derivar o contexto de codificação por frequência ou faixa espectral do segmento atual, por exemplo baseado em 5 coeficientes espectrais vizinhos de segmento anterior e de forma opcional do segmento atual. Em outras palavras, os segmentos podem ser processados no domínio de frequência ou espectral, que pode ser executado por faixa de frequências ou espectral. De forma correspondente, o processador 210 pode, então, ser adaptado para 10 derivar um contexto específico de faixa espectral ou de frequência.

O decodificador de entropia 200 pode ser adaptado para decodificação de entropia do coeficiente atual baseado em uma entropia ou regra de codificação de comprimento variável.

15 ~~O processador 210 pode ser adaptado para derivar o contexto de codificação baseado em um downsampling de frequência~~
ou coeficientes espectrais de um segmento anterior quando o segmento anterior compreende mais coeficientes por largura de banda (ou seja, uma resolução de frequência ou espectral melhor 20 ajustada) do que o segmento atual. Em outras materializações, o processador 210 e o codificador de entropia 220 podem ser configurados para operar baseados em um upsampling de coeficientes espectrais de um segmento anterior, quando o segmento anterior compreender menos coeficientes por largura de banda (ou seja, uma 25 resolução de frequência ou espectral grosseira) do que o segmento atual.

Consequentemente, materializações podem fornecer um método para decodificar um fluxo de áudio codificado para obter

segmentos de coeficientes representando amostras de áudio decodificadas. O método para decodificação pode compreender uma etapa de derivar um contexto de codificação para um coeficiente atual ou atualmente decodificado de um segmento atual baseado em um coeficiente anterior ou anteriormente decodificado, o coeficiente anterior ou anteriormente codificado representando uma resolução diferente de frequência ou tempo do que o coeficiente decodificado atual. Além do mais, o método pode compreender uma etapa de decodificação de entropia do coeficiente atual baseado no contexto de decodificação e o fluxo de áudio codificado. De forma opcional, o método pode compreender uma etapa para determinar os segmentos de coeficientes de áudio codificado do fluxo de áudio codificado, os segmentos representando números diferentes de coeficientes de áudio.

A Fig. 3 ilustra como um processador 110; 210 pode derivar um contexto de codificação para o segmento atual de coeficientes $M_{c,new}$ baseados em um segmento anterior de coeficientes $M_{c,old}$, onde o segmento anterior compreende um número diferente de coeficientes de áudio do que o segmento atual. Na materialização descrita na Fig. 3, o número de coeficientes do segmento M determina a resolução espectral ou de frequência do segmento. A materialização pode compreender um método de mapeamento, que mapeia os coeficientes $M_{c,old}$ de um segmento anterior para coeficientes $M_{c,new}$ tendo a mesma resolução espectral ou de frequência do contexto como segmento atual. Fig. 3 mostra dois conjuntos de coeficientes dentro de dois segmentos, ou seja o segmento anterior original 310 representando coeficientes $M_{c,old}$, $S_{n,0}$, $S_{n,1}$, $S_{n,2}$, etc., e de forma correspondente, o segmento

anterior mapeado 320, que tem maior resolução, ou seja, $M_{c,new}$ é maior do que $M_{c,old}$, representando coeficientes $M_{c,new}$, $S_{n,0}$, $S_{n,1}$, $S_{n,2}$, $S_{n,3}$, etc.

De forma geral, duas materializações podem ser distinguidas, dependendo se a resolução do contexto do segmento atual é maior ou menor que a resolução do contexto do segmento anterior. Fig. 3 ilustra uma materialização, na qual a resolução do segmento anterior dos coeficientes $M_{c,old}$ é menor do que a resolução do segmento atual dos coeficientes $M_{c,new}$. Fig. 3 mostra os coeficientes do segmento anterior 310 e os símbolos para o segmento anterior mapeado 320. Da Fig. 3 pode ser visto que a resolução do segmento atual dos coeficientes $M_{c,new}$ é maior que a resolução do segmento anterior 310 tendo apenas coeficientes $M_{c,old}$.

Em uma materialização o segmento anterior 310 é aumentada a amostra para o segmento 320 de coeficientes $M_{c,new}$ para combinar a resolução espectral ou de frequência do segmento atual. Isso pode incluir upsampling puro com duplicação de símbolo e mecanismos de decimação como por exemplo, repetir cada valor $M_{c,new}$ vezes antes de decimar o segmento com amostra aumentada resultante mantendo apenas 1 coeficiente a cada $M_{c,old}$. Outros mecanismos de interpolação e extrapolação também podem ser usados.

Em materializações, o mapeamento pode ser realizado para todos os segmentos anteriores 310, que são necessários para determinar os contextos para o segmento atual, por exemplo, no tempo n , ou seja, vários segmentos anteriores podem ser levados em conta, ou seja, segmentos anteriores nos tempos $n-1$, $n-2$, etc. Em geral, as materializações podem levar em conta múltiplos períodos de tempo ou segmentos anterior, o número

de slots de tempo necessários para definir um contexto completo pode ser diferente para implementações ou materializações diferentes.

A Fig. 4 ilustra uma outra materialização, em que os coeficientes de um segmento anterior 410 têm as amostras diminuídas para um segmento 420 usado para computar os contextos do segmento atual, ou seja, no qual o número de coeficientes $M_{c,old}$ do segmento anterior 410 é maior que o número de coeficientes $M_{c,new}$ do segmento atual. A Fig. 4 usa uma ilustração semelhante à figura 3, de acordo com a qual coeficientes múltiplos são mostrados em cada segmento 410 e 420. Como ilustrado na figura 4, $M_{c,old}$ é maior que $M_{c,new}$. Portanto os coeficientes $M_{c,old}$ são diminuídos as amostras, para corresponder à resolução espectral ou de frequência do segmento atual de coeficientes $M_{c,new}$, ou seja, em materializações, segmentos anteriores com maior resolução podem ser sub-amostrados para corresponder à resolução do segmento atual com uma resolução mais baixa. Em materializações, podem incluir downsampling puro com duplicação de coeficiente e mecanismos de decimação como, por exemplo, repetir cada valor $M_{c,new}$ vezes antes de decimar o segmento com aumento de amostra resultante mantendo apenas 1 coeficiente a cada $M_{c,old}$. Em outras materializações, as operações do filtro, como por exemplo média de dois ou vários valores adjacentes podem ser levadas em consideração.

A Fig. 5 ilustra uma outra materialização, em que a comutação entre diferentes resoluções é realizada. A Fig. 5 mostra um plano de tempo/frequência, em que três segmentos subsequentes dos coeficientes de áudio são mostrados, ou seja, 510, 520 e 530. Cada um dos segmentos 510, 520 e 530 corresponde a

um único conjunto de coeficientes. Na materialização ilustrada na fig. 5, presume-se que o segundo segmento 520 é duas vezes mais longo que o primeiro e terceiro segmentos 510 e 530. Isto pode ser obtido através de diferentes janelas quando ocorrer segmentação no domínio do tempo, como por exemplo, feito em AAC. Na materialização na fig. 5, presume-se que a taxa de amostragem se mantém constante, em outras palavras, o segundo maior segmento 520 compreende duas vezes mais coeficientes por largura de banda do que o primeiro ou terceiro segmento 510 ou 530.

10 A Fig. 5 mostra que, neste caso, a resolução nas escalas de domínio espectral ou de frequência com a extensão do segmento no domínio de tempo. Em outras palavras, quanto menor a janela no domínio de tempo, mais baixo a resolução no domínio de frequência ou espectral. Ao avaliar os contextos de codificação de amostras no domínio de frequência ou espectral, a Fig. 5 mostra

15 que as necessidades de codificação para ter uma versão de alta resolução do segmento 510 quando ocorrer a codificação do segundo segmento 520, como no exemplo considerado, uma resolução dupla do segmento 510 tem de ser derivada. Em outras materializações, especialmente quando se utiliza outras transformações tempo-frequência ou bancos de filtro, outras relações entre resoluções do domínio de tempo e domínio de frequência podem acontecer.

De acordo com uma materialização, os coeficientes codificados durante o primeiro segmento 510 fornecem uma base para

25 determinar o contexto para o segundo segmento 520, por exemplo, por meio de um upsampling intermediário. Em outras palavras, o conteúdo do contexto proveniente do primeiro segmento 510 pode ser obtido por um up-sampling do primeiro segmento 510, por exemplo,

em termos de interpolação ou extrapolação, a fim de obter o contexto para o segundo segmento 520, tendo uma resolução maior.

Como mostrado na figura. 5, quando comuta-se do segundo segmento 520 para o terceiro segmento 530, o elemento constitutivo do contexto tem que mudar também, pois a resolução já diminuiu. De acordo com uma materialização, os coeficientes codificados durante o segundo segmento 520 podem ser usados a fim de obter o contexto para o terceiro segmento, por meio de um downsampling intermediário. Isto pode ser realizado, por exemplo, em termos de média ou simplesmente usando apenas o cada segundo valor ou outras medidas para downsampling.

Manifestações alcançam a vantagem de uma eficiência aumentada de codificação, levando em conta o contexto passado proveniente de segmentos anteriores, mesmo quando mudanças na resolução ou comprimento da janela ocorreram. Elementos constitutivos de contexto podem ser adaptados para novas resoluções, em termos de upsampling e downsampling, por exemplo, com interpolação e extrapolação, filtragem ou média, etc.

A seguir uma materialização mais específica será apresentada em termos de codificação sem ruído espectral. A codificação espectral silenciosa pode ser usada para reduzir ainda mais a redundância de um espectro quantizado na codificação de áudio. Codificação espectral silenciosa pode ser baseada em uma codificação aritmética em conjunto com uma adaptação de contexto de forma dinâmica.

A codificação silenciosa pode ser baseada em valores espectrais quantizados e podem usar tabelas de frequência cumulativa dependentes de contexto resultantes de, quatro tuplas

vizinhas previamente codificadas. A Fig. 6 ilustra outra materialização. A Fig. 6 mostra um plano de frequência de tempo, onde ao longo do eixo de tempo três horários são indexados N , $N-1$ e $n-2$. Além disso, a Fig. 6 ilustra quatro bandas espectrais ou de frequência que são rotuladas por $m-2$, $m-1$, m e $m+1$. A Fig. 6 mostra dentro de caixa de slot de tempo-frequência, que representam tuplas de amostras a serem codificadas ou decodificadas. Três tipos diferentes de tuplas são ilustradas na fig. 6, em que as caixas com uma borda tracejada ou pontilhada indicam tuplas restantes a serem codificadas ou decodificadas, caixas brancas com uma borda sólida indicam codificação ou decodificação prévia e caixas cinzas com uma borda sólida indicam tuplas previamente decodificadas, que são usadas para determinar o contexto para a tupla atual a ser codificada ou decodificada.

Observe que os segmentos atuais e anteriores referidos nas materializações descritas acima podem corresponder a

uma tupla na presente materialização, em outras palavras, os segmentos podem ser processados na forma de banda no domínio espectral ou de frequência. Como ilustrado na Fig. 6, tuplas ou segmentos na vizinhança da tupla atual (ou seja, no domínio espectral, de tempo ou de frequência) podem ser levadas em consideração para resultar em contexto. Tabelas de frequência cumulativa podem então serem usadas pelo codificador aritmético para gerar um código binário de comprimento variável. O codificador aritmético pode produzir um código binário para um dado conjunto de símbolos e suas respectivas probabilidades. O código binário pode ser gerado mapeando um intervalo de probabilidade, onde o conjunto de símbolos está, para uma palavra-

código. O codificador aritmético pode corresponder a um codificador de entropia 120, respectivamente o codificador de entropia 220 nas materializações mencionadas acima.

Na presente materialização, codificação aritmética baseada em contexto pode ser realizada com base em 4 tuplas (ou seja, em quatro índices de coeficiente espectral, os quais são também rotulados como $q(n,m)$, representando os coeficientes espectrais após quantização, os quais são vizinhos no domínio espectral ou de frequência e qual está codificado por entropia em uma etapa. De acordo com a descrição acima, codificação pode ser realizada baseando-se no contexto de codificação. Como indicado na Fig. 6, de forma adicional para as 4 tuplas, ~~que é codificado~~ (ou seja, o segmento atual) quatro 4 tuplas codificadas previamente são levadas em consideração para derivar o contexto. Essas 4 tuplas determinam o contexto e são prévias na frequência e/ou prévias no domínio de tempo.

A Fig. 7a mostra um fluxograma de uma USAC (USAC = Codificador de Áudio E Fala Universal) codificador aritmético dependente de contexto para o esquema de codificação para o esquema de codificação dos coeficientes espectrais. O processo de codificação depende das 4 tuplas atuais mais o contexto, onde o contexto é usado para selecionar a distribuição de probabilidades do codificador aritmético e para predizer a amplitude dos coeficientes espectrais. Na Fig. 7a a caixa 705 representa determinação de contexto, que é baseada em t_0 , t_1 , t_2 e t_3 correspondendo a $q(n-1, m)$, $q(n, m-1)$, $q(n-1, m-1)$ e $q(n-1, m+1)$, que são as caixas cinzas com bordas sólidas na Fig. 6.

Geralmente, em materializações, o codificador de

entropia pode ser adaptado para codificar o segmento atual nas unidades de 4 tuplas de coeficientes espectrais e para predizer um alcance de amplitude de 4 tuplas baseadas no contexto de codificação.

5 Na presente materialização o esquema de codificação compreende vários estágios. Primeiro, a palavra-código literal é codificada usando um codificador aritmético e uma distribuição específica de probabilidade. A palavra-código representa quatro coeficientes espectrais vizinhos (a,b,c,d),
10 contudo, cada a,b,c,d é limitado no alcance:

$$-5 < a,b,c,d < 4 .$$

 Geralmente, em materializações, o codificador de entropia 120 pode ser adaptada para dividir as 4 tuplas por um fator pré-determinado sempre que necessário para caber o resultado da divisão no alcance predito ou em um alcance pré-determinado e
15 para codificar um número de divisões necessárias, um lembrete de divisão e o resultado da divisão quando as 4 tuplas não está no alcance predito, e para codificação de um lembrete de divisão e o resultado da divisão de outra forma.

20 A seguir, se o termo (a,b,c,d), ou seja qualquer coeficiente a, b, c, d, exceder o dado alcance nessa materialização, isso pode em geral ser considerado por divisão (a,b,c,d) geralmente pelo fator (por exemplo, 2 ou 4) como for necessário, para encaixar o a palavra-código resultante no dado
25 alcance. A divisão pelo fator de 2 corresponde a uma mudança binária para o lado direito, ou seja (a,b,c,d)>> 1. Essa diminuição é feita em uma representação integral, ou seja as informações podem ser perdidas. Os bits menos significantes, que

podem se perder na troca para a direita, são armazenados e mais tarde codificados usando um codificador aritmético e uma distribuição de probabilidade uniforme. O processo de troca para a direita é realizado para todos os quatro coeficientes espectrais

5 (a,b,c,d).

Em materializações gerais, o codificador de entropia 120 pode ser adaptado para codificação do resultado da divisão ou as 4 tuplas usando um índice de grupo ng, o índice de grupo ng referindo-se a um grupo de um ou mais palavras-códigos

10 para quais uma distribuição de probabilidade está baseado no contexto de codificação, e um índice de elemento ne no caso do grupo compreender mais de uma palavra-código, o índice de elemento ne referindo-se a palavra-código dentro do grupo e o índice de elemento pode ser presumido de forma uniformemente distribuída, e

15 para codificação o número de divisões pelo número de símbolos de fuga, um símbolo de fuga sendo um índice de grupo específico ng apenas usado para indicar a divisão e para codificação dos restantes das divisões baseadas na distribuição uniforme usando uma regra de codificação aritmética. O codificador de entropia 120

20 pode ser adaptado para uma codificação de uma sequência de símbolos no fluxo de áudio codificado usando um alfabeto de símbolos compreendendo o símbolo de fuga, e símbolo de grupo correspondentes ao conjunto de índices de grupo disponíveis, um alfabeto de símbolo compreendendo os índices de elemento

25 correspondentes, e um alfabeto de símbolo compreendendo os valores diferentes dos restantes.

Na materialização da Fig. 7a, a distribuição de probabilidade para codificação de palavra-código literal e também

uma estimativa do número de estágios da redução de alcance pode ser derivada de um contexto. Por exemplo, todas as palavras-código, em um total de $8^4 = 4096$, aparecem um total de 544 grupos, que consistem de um ou mais elementos. A palavra-código pode ser representada em um fluxo de bits enquanto o índice de grupo ng e o elemento de grupo ne . Ambos os valores podem ser codificados usando um codificador aritmético, usando certas distribuições de probabilidade. Em uma materialização, a distribuição de probabilidade para ng pode ser obtida do contexto, considerando a distribuição de probabilidade para ne pode ser presumido ser uniforme. Uma combinação de ng e ne pode identificar de forma ambigua uma palavra-código. O restante da divisão, ou seja os planos de bit alternados, podem ser presumido como uniformemente distribuído.

Na Fig. 7a, na etapa 710, as 4 tuplas $q(n,m)$, que são (a,b,c,d) ou o segmento atual é fornecido e um parâmetro lev é iniciado definindo para 0.

Na etapa 715 do contexto, o alcance de (a,b,c,d) está estimado. De acordo com essa estimativa, (a,b,c,d) pode ser reduzido por $lev0$ níveis, ou seja dividido por um fator de 2^{lev0} . Os planos de bit menos significantes de $lev0$ são armazenados para uso posterior na etapa 750.

Na etapa 720 é verificado se (a, b, c, d) excede o alcance determinada e em caso afirmativo, o intervalo de (a, b, c, d) é reduzido por um fator de 4 na etapa 725. Em outras palavras, na etapa 725 (a, b, c, d) são modificadas por 2 para a direita e os planos de bit removidos são armazenados para posterior utilização na etapa 750.

Para indicar esta etapa de redução, ng está definido para 544 na etapa 730, ou seja $ng = 544$ serve como uma palavra-código de fuga. Esta palavra-código é, então, escrito ao fluxo de bits na etapa 755, onde para derivar a palavra-código na

5 etapa 730 um codificador aritmético com uma distribuição de probabilidade derivada a partir do contexto é usado. Caso essa etapa de redução seja aplicado a primeira vez, ou seja se $lev == lev0$, o contexto é ligeiramente adaptado. Caso a etapa de redução seja aplicado mais de uma vez, o contexto é descartado e

10 uma distribuição padrão é usada adiante. O processo então continua com etapa 720.

Se na etapa 720 uma combinação para o intervalo for detectada, mais especificamente, se $(\bar{a}, \bar{b}, \bar{c}, \bar{d})$ correspondem à condição de alcance, o (a, b, c, d) é mapeado para um grupo ng ,

15 e, se aplicável, o índice de elemento de grupo ne . Esse mapeamento não é ambíguo, isto é, (a, b, c, d) pode ser obtido do ng e ne . O índice do grupo ng é então codificado pelo codificador aritmético, utilizando uma distribuição de probabilidades utilizada para o contexto adaptado/descartado na etapa 735. O índice do grupo ng é

20 então inserido no fluxo de bits na etapa 755. Em uma etapa seguinte 740, é verificado que se o número de elementos do grupo é maior que 1. Se necessário, isto é, se os grupos indexados por ng consistem em mais de um elemento, o índice de elemento de grupo ne é codificado pelo codificador aritmético na etapa 745, assumindo

25 uma distribuição uniforme na presente materialização.

Seguindo a etapa 745, o índice de grupo de elemento ne é inserido no fluxo de bits na etapa 755. Finalmente, na etapa 750, todos os planos de bit são codificados usando

codificador aritmético, assumindo uma distribuição uniforme de probabilidade. Os planos de bit codificados armazenados são então inseridos no fluxo de bits na etapa 755.

Em materializações, o codificador de entropia 220
 5 pode ser adaptado para decodificação de um índice do grupo ng do fluxo de áudio codificado com base em uma distribuição de probabilidade derivada a partir do contexto de codificação, onde o índice do grupo ng representa um grupo de uma ou mais palavras-código, e, com base em uma distribuição de probabilidade uniforme,
 10 para decodificar um índice do elemento ne do fluxo de áudio codificado se o índice do grupo ng indica um grupo composto por mais de uma palavra-código, e definir 4 tuplas de coeficientes espectrais do segmento atual com base no índice do grupo ng e o índice do elemento ne, obtendo assim a representação do domínio
 15 spectral nas tuplas de coeficientes espectrais.

O decodificador de áudio 220 pode ser adaptado
 para decodificação de uma sequência de símbolos do fluxo de áudio codificado com base em uma distribuição de probabilidade derivada a partir do contexto de codificação usando um alfabeto de símbolo
 20 compreendendo um símbolo de fuga e símbolos de grupo correspondentes a um conjunto de índices disponíveis de grupo ng, para obter quatro tuplas preliminares de coeficientes espectrais baseados em um índice disponível de grupo ng para qual um símbolo de grupo da sequência de símbolos correspondentes, e baseado no
 25 índice de elemento ne, e para multiplicar as 4 tuplas preliminares com o fator dependendo do número de símbolos de fuga na sequência dos símbolos para obter a tupla dos coeficientes espectrais.

O decodificador de áudio 220 pode ser adaptado

para decodificação de restante de divisão do fluxo de áudio codificado baseado em uma distribuição de probabilidade uniforme usando uma regra de codificação aritmética e para adição do restante às 4 tuplas preliminares multiplicadas para obter as 4
5 tuplas dos coeficientes espectrais.

O decodificador de entropia 220 pode ser adaptado para multiplicar as 4 tuplas com um fator pré-determinado, bem como quando um símbolo de fuga é decodificado do fluxo de áudio codificado, um símbolo de fuga sendo um índice de grupo ng específico apenas usado para indicar uma multiplicação, e para a
10 decodificação de um restante do fluxo de áudio codificado com base em uma distribuição de probabilidade uniforme utilizando uma regra de codificação aritmética, o decodificador de entropia 220 ainda pode ser adaptado para adicionar o restante ao multiplicado, as 4
15 tuplas para obter o segmento atual.

Na sequência de uma materialização de um codificador aritmético USAC dependente do contexto o regime de decodificação será descrito. Correspondente à materialização acima do esquema de codificação, 4 tuplas correspondentes aos
20 coeficientes espectrais quantizados, que são codificados sem ruído, são considerados. Além disso, supõe-se que as 4 tuplas são transmitidas a partir da menor frequência ou coeficiente espectral e progredindo para a maior frequência ou coeficientes espectrais. Os coeficientes podem, por exemplo, corresponder aos coeficientes
25 AAC, que são armazenados em uma matriz e, a ordem de transmissão das palavras-código de codificação sem ruído é suposto ser de tal forma que quando elas são decodificadas na ordem recebida e armazenada no *array*, *bin* é o índice mais rapidamente incrementável

e g é o índice mais lentamente incrementável. Dentro de uma palavra-chave, a ordem de decodificação é a, b, c, d.

A Fig. 7-B ilustra o procedimento de atualização do contexto geral de uma materialização. Os detalhes sobre o contexto de adaptação de acordo com um mecanismo de previsão de profundidade de bits serão considerados na presente materialização. A Fig. 7b mostra um plano 760 exibindo a possível gama de 4 tuplas (a, b, c, d) em termos de planos de bits. A profundidade de bits, ou seja, número de planos de bits necessários para representar 4 tuplas, pode ser prevista pelo contexto das atuais 4 tuplas, por meio do cálculo da variável chamada lev0, que também é indicada na figura 7b. As 4 tuplas são então divididas por $2^{\text{lev}0}$, ou seja lev=lev0 planos de bits são retirados e armazenados para uso posterior de acordo com a etapa 715 acima descrito.

Se as 4 tuplas estão no intervalo $-5 < a, b, c, d < 4$, a profundidade de bit lev0 predita foi corretamente predita ou superestimada. As 4 tuplas podem ser então codificadas pelo índice de grupo ng, o índice de elemento ne e os planos de bits lev restantes, em conformidade com a descrição acima. A codificação das 4 tuplas atuais então é finalizada. A codificação do índice de elemento ne é indicada na figura 7b pela distribuição de probabilidade uniforme 762, que será a seguir sempre utilizado para os índices de elemento de codificação, que na fig. 7B o parâmetro r representa o restante das 4 tuplas após a divisão e p(r) representa a função de densidade de probabilidade uniforme correspondente.

Se as 4 tuplas não estão no intervalo $-5 < a, b,$

c, $d < 4$, a predição com base no contexto de codificação 764 é muito baixa, um símbolo de fuga ($ng = 544$) é codificado 766 e as 4 tuplas estão divididas por 4 e o lev incrementado por 2, conforme a etapa 730 na fig. 7a. O contexto é adaptado como segue. Se
 5 $lev == lev0 + 2$ o contexto é ligeiramente adaptado, correspondendo à 768 na Fig. 7b. Uma bandeira pode ser definida na representação do contexto, t , e um novo modelo de distribuição de probabilidade então será usado para codificação de símbolos ng futuros.

Se $lev > lev0$ outro símbolo de fuga é codificado
 10 de acordo com a etapa 770 na fig. 7b o contexto é completamente redefinido, cf. 772, descartados como na etapa 730 na fig. 7a, respectivamente. Nenhuma adaptação do contexto é utilizada ainda
 porque não é considerada relevante para a atual codificação das 4
 tuplas. O modelo atual de probabilidade, o utilizado quando nenhum
 15 contexto está disponível, é então usado para os futuros símbolos
 ~~ng , que é indicado pelas etapas 774 e 776 na fig. 7b. O processo é~~
 então repetido para outras tuplas.

Para resumir, a adaptação de contexto é um mecanismo a fim de reduzir o significado de contexto na
 20 codificação adaptativa de contexto. A adaptação de contexto pode ser acionada quando a $lev0$ predita e o lev real não combinam. Ela é facilmente detectada pelo número de símbolos codificados de fuga ($ng = 544$), cf. 766 e 770 na fig. 7b, e portanto pode ser realizada no decodificador de forma semelhante também.

25 A adaptação do contexto pode ser feita desencadeando uma bandeira na representação do estado de contexto t . O valor t é calculado pela função `get_state()`, como $lev0$, usando o contexto derivado da estrutura anterior e/ou atual ou

segmento das 4 tuplas atuais, que são armazenadas em uma tabela q
 [][]. O estado do contexto pode, por exemplo, ser representado por
 24 bits. Existem 1905800 estados possíveis em uma materialização.
 Esses estados podem ser representados por apenas 21 bits. Os bits
 5 23 e 24 de t são reservados para a adaptação do estado do
 contexto. De acordo com os valores dos bits 23 e 24, o get_pk()
 produzirá modelos diferentes de distribuição de probabilidade. Em
 uma materialização, o bit 23 de t pode ser definido como um,
 quando as 4 tuplas são divididas por 4, depois de previamente
 10 divididas por lev0, ou seja, lev==lev0+2.

Em consequência, o mapeamento entre o estado de
 contexto t e o modelo de distribuição de probabilidade pki, é
 diferente para lev==lev0+2 do que para lev==lev0. O mapeamento
 entre o estado de contexto t e o modelo pki é pré-definido durante
 15 a fase de formação através da realização de otimizações nas
 estatísticas gerais da sequência de treinamento. Quando
 lev>lev0+2, o contexto e t pode ser definido como zero. As saídas
 Get_pk(), então o modelo padrão de distribuição de probabilidade
 pki, o que corresponde a t=0.

20 A seguir, os detalhes de um mapeamento de
 contexto em uma materialização serão descritas. O mapeamento do
 contexto é a primeira operação feita na codificação adaptativa de
 contexto após a redefinição eventual do contexto na presente
 materialização. É feito em duas etapas.
 25 Primeiro, antes da codificação, a tabela de contexto qs[] de
 tamanho previous_lg / 4, salva na estrutura anterior, é mapeada em
 uma tabela de contexto q[0][] de tamanho lg / 4 correspondente ao
 tamanho da estrutura atual. O mapeamento é feito na função

arith_map_context, o que é exemplificado pelo seguinte pseudo-código:

```

/*input variable*/
lg/4 /*number of 4-tuples*/
5   arith_map_context()
    {
        v=w=0

        if(core_mode==1){
            q[0][v++]=qs[w++];
10   }

        ratio= ((float)previous_lg)/((float)lg);
        for(j=0; j<lg/4; j++){
            k= (int)((float)((j)*ratio));
            q[0][v++] = qs[w+k];
15   }

        if(core_mode==0){
            q[0][lg/4]=qs[previous_lg/4];
        }
            q[0][lg/4+1]=qs[previous_lg/4+1];
20   previous_lg=lg;
        }

```

Como pode ser visto a partir do pseudo-código, o mapeamento pode não ser exatamente o mesmo para todas as estratégias de codificação. Na presente materialização, o mapeamento se difere quando AAC (Codificação de Áudio Avançada) é usado (core_mode == 0) para um coeficiente de quando TCX (base de Codificação de Transformada) é usado (core_mode == 1). Uma das diferenças está no modo como os limites das tabelas são

manipulados. Em AAC, o mapeamento pode começar a partir do índice 0 (primeiro valor da tabela), enquanto para TCX pode iniciar a partir do índice 1 (segundo valor na tabela), sabendo que o primeiro valor é sempre definido como "desconhecido" (estado específico utilizado para redefinir o contexto). A proporção de `previous_lg` sobre `lg` irá determinar a ordem do upsampling ($\text{proporção} < 1$) ou o downsampling ($\text{proporção} > 1$), que será realizado na presente materialização. A Fig. 7c ilustra o caso do TCX na conversão de uma tabela de contexto salva de tamanho 1024 / 4, cf. lado esquerdo 780 da Fig. 7c, para um tamanho de 512/4, cf. lado direito 782 da Fig. 7c. Pode-se observar que, enquanto para a tabela de contexto atual 782 incrementos de etapas 1 são utilizados, para a tabela de contexto salva 780 incrementos de etapa da proporção descrita acima são utilizados.

A Fig. 7c ilustra o procedimento de atualização de contexto de uma materialização para alterações de resolução.

Uma vez que o mapeamento é feito, a codificação adaptativa de contexto é realizada. No final da codificação, os elementos da estrutura atual são salvos na tabela `qs[]` para a próxima estrutura. Pode ser feito em `arith_update_context()`, que é exemplificado pelo seguinte pseudo-código:

```
/*input variables*/
a,b,c,d /* value of the decoded 4-tuple */
i /*the index of the 4-tuple to decode in the
vector*/

lg/4 /*number of 4-tuples*/
arith_update_context()
{
```

```

        q[1][1+i].a=a;
        q[1][1+i].b=b;
        q[1][1+i].c=c;
        q[1][1+i].d=d;
5         if ( (a<-4) || (a>=4) || (b<-4) || (b>=4) || (c<-
            4) || (c>=4) || (d<-4) || (d>=4) ) {
            q[1][1+i].v =1024;
        }
        else q[1][1+i].v=egroups[4+a][4+b][4+c][4+d];
10         if(i==lg/4 && core_mode==1){
            qs[0]=q[1][0];
            ratio= ((float) lg)/((float)1024);
            for(j=0; j<256; j++){
                k = (int) ((float) j*ratio);
15         qs[1+k] = q[1][1+j] ;
            }
        }

        qs[previous_lg/4+1] = q[1][lg/4+1];
        previous_lg = 1024;
    }
20     if(i==lg/4 && core_mode==0){
        for(j=0; j<258; j++){
            qs[j] = q[1][k];
        }
        previous_lg = min(1024,lg);
25     }
    }

```

Na presente materialização, o salvamento é feito de maneira diferente de acordo com o programador principal (AAC ou

TCX). Em TCX o contexto é sempre salvo em uma tabela `qs[]` de valores 1024/4. Esse mapeamento adicional pode ser feito por causa da decisão de malha fechada da AMR-WB + (Adaptive Multirate WideBand Codec). Na decisão de malha fechada muitos procedimentos de cópia de estados do codificador são necessários para testar cada combinação possível de TCXs e ACELP (Arithmetic Coded Excited Linear Prediction). A cópia do estado é mais fácil de implementar quando todos os modos TCX compartilham o mesmo tamanho para a tabela `qs[]`. Um mapeamento é, então, usado para converter de forma sistemática a partir de lg/4 para 1024/4. Por outro lado, AAC salva somente o contexto e não executa o mapeamento durante esta fase.

A Fig. 8 ilustra um fluxograma da materialização do esquema de decodificação. Na etapa 805, correspondente à etapa 705, o contexto é derivado com base em `T0`, `T1`, `T2` e `T3`. Na etapa 810, o primeiro nível de redução `lev0` é estimado a partir do contexto, e o leve variável é definido como `lev0`. Na etapa seguinte 815, o grupo NG é lido a partir do fluxo de bits e a distribuição de probabilidade para a decodificação de `ng` é derivado do contexto. Na etapa 815, o grupo `ng` pode então ser decodificado a partir do fluxo de dados. Na etapa 820 é determinado se o `ng` é igual a 544, que corresponde ao valor de fuga. Se assim for, o variável `lev` pode ser aumentada em 2 antes de voltar para a etapa 815. Caso essa seção seja usada pela primeira vez, ou seja, se `lev==lev0`, a distribuição de probabilidade, respectivamente, o contexto pode ser adaptado em conformidade, respectivamente descartada se a seção não for usada pela primeira vez, em consonância com o mecanismo de adaptação de

contexto acima descrito, cf. Figs. 7B e 7C. No caso do índice do grupo ng não ser igual a 544 na etapa 820, em uma etapa seguinte 825 é determinado se o número de elementos de um grupo é maior que 1, e sendo assim, na etapa 830, o elemento do grupo ne é lido e decodificado a partir do fluxo de bits, assumindo uma distribuição de probabilidade uniforme. O índice do elemento ne é derivado do fluxo de bits usando codificação aritmética e uma distribuição de probabilidade uniforme.

Na etapa 835 a palavra-código literal (a , b , c , d) é derivada de ng e ne , por exemplo, por um processo de visualização nas tabelas, por exemplo, referem-se a $dgroups[ng]$ e $acod_ne[ne]$.

Na etapa 840 para todos os planos de bit sem lev, os planos são lidos a partir do fluxo de bits usando codificação aritmética e assumindo uma distribuição de probabilidade uniforme. Os planos de bits podem então ser anexados a (a , b , c , d) trocando

(a , b , c , d) para a esquerda e acrescentando o plano de bit bp :
 $((a, b, c, d) \ll 1) \mid = bp$. Este processo pode ser repetido lev vezes.

Finalmente na etapa 845 as 4 tuplas $q(n, m)$, ou seja, (a , b , c , d) podem ser fornecidas.

A seguir, pseudocódigos e detalhes de implementação são fornecidos de acordo com uma materialização. As definições a seguir serão utilizadas.

(a, b, c, d) 4 tuplas para decodificar
 ng Índice de grupo do plano de 2 bits mais
 significante das 4 tuplas, onde $0 \leq ng \leq 544$. O último valor 544
 corresponde ao símbolo de fuga, ARITH_ESCAPE.

ne Índice de elemento dentro do grupo. ne está entre 0 e o cardinal de cada grupo mm. O número máximo de elementos dentro de um grupo é 73.

5 lev Nível dos planos de bits restantes. Corresponde ao número de planos de bits menos significantes do que os mais significantes planos de 2 bits.

egroups Tabela de índice de grupo [a][b][c][d]. Permite mapear o plano de 2 bits mais significante das 4 tuplas
10 (a,b,c,d) nos 544 grupos.

mm Cardinal do grupo

og Offset do grupo

~~dggroups[] Mapeia o índice de grupo ng para o cardinal de cada grupo mm (primeiros 8 bits) e o offset do grupo
15 og em dgvector[] (últimos 8 bits).~~

~~dgvector[] Mapeia o offset do grupo og e o índice do elemento ne para o plano de 2 bits mais significante das 4 tuplas (a,b,c,d).~~

arith_cf_ng_hash[] Tabela hash mapeando o estado
20 do contexto para um índice cumulativo de tabela de frequências pki.

arith_cf_ng[pki][545] Modelos das frequências cumulativas para o símbolo do índice de grupo ng.

arith_cf_ne[] Frequências cumulativas para
25 o símbolo de índice do elemento ne.

r Plano de bit das 4 tuplas menos significantes do que o plano de 2 bits mais significante.

arith_cf_r [] Frequências cumulativas para o

símbolo dos planos de bits ne menos significantes.

A seguir, o processo de decodificação é considerado primeiro. Os coeficientes espectrais quantizados de 4 tuplas são silenciosamente codificados e transmitidos a partir da menor frequência ou o coeficiente espectral e progredindo para a maior frequência ou coeficiente espectral. Os coeficientes da AAC são armazenados na matriz `x_ac_quant[g][win][sfb][bin]`, e a ordem de envio das palavras-código de codificação sem ruído é tal que, quando eles são decodificados na ordem recebida e armazenada na matriz, `bin` é o índice que aumenta mais rápido e `g` é o índice mais lentamente incrementado. Dentro de uma palavra-código, a ordem de decodificação é a, b, c, d. O coeficiente do TCX é armazenado diretamente na matriz `x_tcx_invquant[win][bin]`, e a ordem de transmissão das palavras-código de codificação sem ruído é tal que, quando eles são decodificados na ordem recebida e armazenada na matriz, `Bin` é o índice mais rapidamente incrementado e `win` é o índice mais lentamente incrementado. Dentro de uma palavra-código da ordem de decodificação é a, b, c, d. Primeiro, a bandeira `arith_reset_flag` determina se o contexto deve ser redefinido. Se a bandeira for VERDADEIRA a função a seguir é chamada:

```

*global variables*/
q[2][290] /*current context*/
qs[258] /*past context*/
previous_lg /*number of 4-tuples of the past
context*/

arith_reset_context()
{
    for(i=0;i<258;i++){

```

```

        qs[i].a=0; qs[i].b=0; qs[i].c=0; qs[i].d=0
        qs[i].v=-1;
    }
    for(i=0;i<290;i++){
5      q[0][i].a=0;          q[0][i].b=0;          q[0][i].c=0;
        q[0][i].d=0

        q[0][i].v=-1;
        q[1][i].a=0;          q[1][i].b=0;          q[1][i].c=0;
        q[1][i].d=0
10     q[1][i].v=-1;
    }

    previous_lg=256;
    }

```

Caso contrário, quando a arith_reset_flag for

15 FALSA, o mapeamento é feito entre o contexto passado e do contexto

atual:

```

        /*input variable*/
        lg /*number of 4-tuples*/
        arith_map_context(lg)
20     {
        v=w=0
        if(core_mode==1){
        q[0][v++]=qs[w++];
        }
25     ratio= ((float)previous_lg)/((float)lg);
        for(j=0; j<lg; j++){
        k = (int) ((float)) ((j)*ratio);
        q[0][v++] = qs[w+k];

```

```

    }
    if(core_mode==0){
    q[0][lg]=qs[previous_lg];
    }
5    q[0][lg+1]=qs[previous_lg+1];
    previous_lg=lg;
    }

```

O decodificador sem ruído emite 4 tuplas de coeficientes espectrais quantizados assinados. No começo, o estado do contexto é calculado com base em quatro grupos previamente decodificados em torno de 4 tuplas para decodificar. O estado do contexto é dado pela função `arith_get_context ()`:

```

/*input variables*/
i /*the index of the 4-tuple to decode in the
15 vector*/
arith_get_context(i,)
{
    t0=q[0][1+i].v+1;
    t1=q[1][1+i-1].v+1;
20    t2=q[0][1+i-1].v+1;
    t3=q[0][1+i+1].v+1;
    if ( (t0<10) && (t1<10) && (t2<10) && (t3<10) ){
    if ( t2>1 ) t2=2;
    if ( t3>1 ) t3=2;
25    return 3*(3*(3*(3*(3*(10*(10*t0+t1))+t2)+t3)));
    }
    if ( (t0<34) && (t1<34) && (t2<34) && (t3<34) ){
    if ( (t2>1) && (t2<10) ) t2=2; else if ( t2>=10 )

```


t2=3;

if ((t3>1) && (t3<10)) t3=2; else if (t3>=10)
t3=3;

5 return 252000+4*(4*(34*(34*t0+t1))+t2)+t3;
 }

if ((t0<90) && (t1<90)) return
880864+90*(90*t0+t1);

if ((t0<544) && (t1<544)) return 1609864 +
544*t0+t1;

10 if (t0>1)
 {
 a0=q[0][i].a;

b0=q[0][i].b;

c0=q[0][i].c;

15 d0=q[0][i].d;

}

else a0=b0=c0=d0=0;

if (t1>1)

{

20 a1=q[1][i-1].a;

b1=q[1][i-1].b;

c1=q[1][i-1].c;

d1=q[1][i-1].d;

}

25 else a1=b1=c1=d1=0;

l=0;

do

{

```

        a0>>=1;
        b0>>=1;
        c0>>=1;
        d0>>=1;
5      a1>>=1;
        b1>>=1;
        c1>>=1;
        d1>>=1;
        l++;
10     }

        while ( (a0<-4) || (a0>=4) || (b0<-4) || (b0>=4)
|| (c0<-4) || (c0>=4) || (d0<-4) || (d0>=4) ||
15     (a1<-4) || (a1>=4) || (b1<-4) || (b1>=4) || (c1<-
4) || (c1>=4) || (d1<-4) || (d1>=4) );

        if ( t0>1 ) t0=1+(egroups[4+a0][4+b0][4+c0][4+d0]
>> 16);

        if ( t1>1 ) t1=1+(egroups[4+a1][4+b1][4+c1][4+d1]
>> 16);

        return 1609864 + ((l<<24)|(544*t0+t1));
20     }

    Uma vez que o estado é conhecido o grupo a quem
pertence o plano de 2 bits mais significativo de 4 tuplas é
decodificado usando o arith_decode () alimentado com a tabela
apropriada de frequências cumulativas correspondente ao estado de
25 contexto. A correspondência é feita pela função arith_get_pk ):

    /*input variable*/

    s /* State of the context*/

    arith_get_pk(s)

```

```

{
    psci[28] = {
        247,248,249,250,251,252,253,254,254,0,254,254,254
    ,255,250,215,
5      215,70,70,123,123,123,123,3,67,78,82,152
    };
    register unsigned char *p;
    register unsigned long i,j;
    i=123*s;
10     for (;;)
    {
        j=arith_cf_nq_hash[i&32767];
        if ( j==0xFFFFFFFFul ) break;
        if ( (j>>8)==s ) return j&255;
15     i++;
    }

    p=psci+7*(s>>22);
    j= s & 4194303;
    if ( j<436961 )
20     {
        if ( j<252001 ) return p[(j<243001)?0:1]; else
    return p[(j<288993)?2:3];
    }
    else
25     {
        if ( j<1609865 ) return p[(j<880865)?4:5]; else
    return p[6];
    }
}

```

```

}
```

Então a função `arith_decode()` é chamada com a tabela de frequências cumulativas, correspondente ao retorno do índice pelo `arith_get_pk ()`. O codificador aritmético é uma
 5 implementação inteira gerando tag com escalonamento. O seguinte pseudo-código C descreve o algoritmo utilizado.

```

/*helper funtions*/
```

```

bool arith_first_symbol(void);
```

```

/* Return TRUE if it is the first symbol of the
```

```

10 sequence, FALSE otherwise*/
```

```

Ushort arith_get_next_bit(void);
```

```

/* Get the next bit of the bitstream*/
```

```

/* global variables */
```

```

low
```

```

15 high
```

```

value
```

```

/* input variables */
```

```

cum_freq[];
```

```

cfl;
```

```

20 arith_decode()
```

```

{
```

```

if(arith_first_symbol())
```

```

{
```

```

value = 0;
```

```

25 for (i=1; i<=20; i++)
```

```

{
```

```

value = (val<<1) | arith_get_next_bit();
```

```

}
```

```

        low=0;
        high=1048575;
    }
    range = high-low+1;
5      cum    =((((int64)    (value-low+1))<<16)-((int64)
1)))/((int64) range);

        p = cum_freq-1;
        do
        {
10      q=p+(cfl>>1);

        if ( *q > cum ) { p=q; cfl++; }

        cfl>>=1;
        }

        while ( cfl>1 );

-15  symbol-= p-cum_freq+1;

        if(symbol)

            high          =          low          +

            (((int64)range)*((int64)cum_freq[symbol-1]))>>16 - 1;

            low    +=          (((int64)    range)*          ((int64)
20  cum_freq[symbol]))>>16;

        for (;;)
        {

            if ( high<524286) { }

            else if ( low>=524286)

25      {

            value -=524286;

            low -=524286;

            high -=524286;

```

```

    }
    else if ( low>=262143 && high<786429)
    {
        value -= 262143;
5       low -= 262143;
        high -= 262143;
    }
    else break;
    low += low;
10    high += high+1;
    value = (value<<1) | arith_get_next_bit();
    }
    return symbol;
}

```

15 ~~Enquanto o índice de grupo decodificado ng é o~~
~~símbolo de fuga, ARITH_ESCAPE, um índice grupo adicional ng é~~
 decodificado e a variável lev é incrementada por dois. Uma vez que
 o índice do grupo decodificado não é o símbolo de fuga,
 ARITH_ESCAPE, o número de elementos, mm, dentro do grupo e o
 20 offset de grupo, og, são deduzidos através da observação da tabela
 dgroups[]:

```

mm=dgroups[nq]&255
og = dgroups[nq]>>8

```

O índice do elemento ne é, então, decodificado
 25 chamando o arith_decode() com a tabela de frequências cumulativas
 (arith_cf_ne + ((mm * (mm-1))>> 1) []). Uma vez que o índice do
 elemento é decodificado, o plano de 2 bits mais significativo das
 4 tuplas pode ser obtido com a tabela dgvector[]:

```

a=dgvectors[4*(og+ne)]
b=dgvectors[4*(og+ne)+1]
c=dgvectors[4*(og+ne)+2]
d=dgvectors[4*(og+ne)+3]

```

5 Os planos de bits restantes são, então, decodificados a partir do nível mais significativo para o menos significativo chamando lev vezes o `arith_decode()` com a tabela de frequências cumulativas `arith_cf_r []`. O plano de bit decodificado `r` permite refinar as 4 tuplas da seguinte forma:

```

10      a = (a<<1) | (r&1)
      b = (b<<1) | ((r>>1)&1)
      c = (c<<1) | ((r>>2)&1)
      d = (d<<1) | (r>>3)

```

15 Uma vez que as 4 tuplas `(a, b, c, d)` são completamente decodificadas as tabelas de contexto `q` e `qs` são atualizadas chamando a função `arith_update_context ()`.

```

      arith_update_context(a,b,c,d,i,lg)
      {
20      q[1][1+i].a=a;
      q[1][1+i].b=b;
      q[1][1+i].c=c;
      q[1][1+i].d=d;
      if ( (a<-4) || (a>=4) || (b<-4) || (b>=4) || (c<-
25  4) || (c>=4)
      || (d<-4) || (d>=4) )
      q[1][1+i].v =1024;
      else q[1][1+i].v=egroups[4+a][4+b][4+c][4+d];

```

```

        if(i==lg && core_mode==1){
            qs[0]=q[1][0];
            ratio= ((float) lg)/((float)256);
            for(j=0; j<256; j++){
5              k = (int) ((float)) ((j)*ratio);
              qs[l+k] = q[1][l+j] ;
            }
            qs[previous_lg+1]=q[1][lg+1];
            previous_lg=256;
10          }
            if(i==lg && core_mode==0){
                for(j=0; j<258; j++){
                    qs[j] = q[1][k];
                }
15          previous_lg=min(1024,lg);
        }
    }
}

```

Dependendo de certos requisitos de implementação dos métodos inventivos, os métodos inventivos podem ser implementados em hardware ou software. A implementação pode ser formada usando um meio de armazenamento digital, em particular, um disco, um DVD ou um CD, com um sinal de controle de leitura eletrônica nele, que coopera com o computador programável, de modo que os métodos inventivos são executados. Geralmente, a presente invenção é, portanto, um produto de programa de computador com um código de programa para um portador de leitura óptica, o código do programa sendo operativo para a realização dos métodos inventivos quando o programa de computador é executado em um computador. Em

outras palavras, os métodos inventivos são, portanto, um programa de computador com um código de programa para realizar pelo menos um dos métodos inventivos quando o programa de computador for executado em um computador.

REIVINDICAÇÕES

1. "CODIFICADOR DE ÁUDIO E DECODIFICADOR DE ÁUDIO", **caracterizado** pelo codificador de áudio (100) para codificar uma sequência de segmentos de coeficientes, os segmentos
5 sendo subsequentes um ao outro no tempo compreender:

meios para fornecer a sequência de segmentos dos coeficientes de um fluxo de áudio representando um sinal de áudio amostrado pelo uso de comprimentos diferentes de transformada, tais quais segmentos de coeficientes para os quais comprimentos
10 diferentes de transformada são usados, representa espectralmente o sinal de áudio amostrado em resoluções diferentes de frequência;

um processador (110) para obter um contexto de codificação para um coeficiente codificado de um segmento atual com base em um coeficiente previamente codificado de um segmento
15 anterior, os segmentos anteriores e atuais correspondentes a resoluções diferentes de frequência e comprimentos diferentes de transformada, respectivamente; e

um codificador de entropia 120 para codificação de entropia do atual coeficiente baseado no contexto de
20 codificação para obter um fluxo de áudio codificado.

2. Codificador de áudio (100), de acordo com a da reivindicação 1, **caracterizado** pelo codificador de entropia (120) ser adaptado para codificar o coeficiente atual em unidades de uma tupla dos coeficientes espectrais e para prever um
25 alcance da tupla baseada no contexto de codificação.

3. Codificador de áudio (100), de acordo com a reivindicação 2, **caracterizado** pelo codificador de entropia (120) ser adaptado para dividir a tupla por um fator pré-determinado

quando necessário para encaixar um resultado da divisão em um alcance pré-determinado e para codificar um número de divisões necessárias, um lembrete de divisão e o resultado da divisão quando as tuplas não estão no alcance predito, e para codificação
5 de um lembrete de divisão e o resultado da divisão de outra forma.

4. Codificador de áudio (100), de acordo com a reivindicação 3, **caracterizado** pelo codificador de entropia (120) ser adaptado para codificação do resultado da divisão ou as tuplas usando um índice de grupo, o índice de grupo referindo-se a um
10 grupo de um ou mais palavras-códigos, para as quais uma distribuição de probabilidade está baseado no contexto de codificação, e baseado em uma distribuição de probabilidade uniforme, um índice de elemento no caso do grupo compreender mais
de uma palavra-código, o índice de elemento referindo-se a
15 palavra-código dentro do grupo e para codificação o número de divisões pelo número de símbolos de fuga, um símbolo de fuga sendo
um índice de grupo específico apenas usado para indicar a divisão e para codificação dos restantes das divisões baseadas na distribuição uniforme usando uma regra de codificação aritmética.

20 5. Codificador de áudio (100), de acordo com a reivindicação 4, **caracterizado** pelo codificador de entropia (120) ser adaptado para codificação de uma sequência de símbolos no fluxo de áudio codificado usando um alfabeto de símbolos compreendendo o símbolo de fuga, um símbolos de grupo
25 correspondentes ao conjunto de índices de grupo disponíveis, um alfabeto de símbolo compreendendo os índices de elemento correspondentes, e um alfabeto de símbolo compreendendo os valores diferentes dos restantes.

6. Codificador de áudio (100), de acordo com a uma das reivindicações de 1 a 5, **caracterizado** pelo processador (110) e o codificador de entropia (120) serem configurados para operar baseados em um downsampling de coeficientes espectrais dos
5 segmentos anteriores, quando o segmento anterior exibe uma melhor resolução espectral do que o segmento atual e/ou onde o processador (110) e o codificador de entropia (120) estão configurados para operar baseados em um upsampling de coeficientes espectrais do segmento anterior, quando o segmento anterior exibe
10 uma resolução espectral mais rígida que o segmento atual.

7. Método para codificação de uma sequência de segmentos de coeficientes, **caracterizado** pelos segmentos sendo subseqüentes para cada tempo, compreenderem os passos de:

fornece a sequência de segmentos dos
15 coeficientes de um fluxo de áudio representando um sinal de áudio amostrado pelo uso de comprimentos diferentes de transformada,

tais quais segmentos de coeficientes para os quais comprimentos diferentes de transformada são usados, representa espectralmente o sinal de áudio amostrado em resoluções diferentes de frequência;

20 obter um contexto de codificação para um coeficiente codificado de um segmento atual com base em um coeficiente previamente codificado de um segmento anterior, os segmentos anteriores e atuais correspondentes a resoluções diferentes de frequência e comprimentos diferentes de
25 transformada, respectivamente; e

codificação de entropia do atual coeficiente baseado no contexto de codificação para obter um fluxo de áudio codificado.

8. Codificador de áudio (100) para os segmentos de codificação de coeficientes, **caracterizado** pelos segmentos de coeficientes representando diferentes resoluções de tempo ou frequência de um sinal de áudio amostrado, compreender:

5 um meio para fornecer os segmentos dos coeficientes de um fluxo de áudio, os coeficientes formando uma representação espectral de um sinal de áudio representado por um fluxo de áudio em uma resolução espectral variando entre os segmentos;

10 um processador 110 para obter um contexto de codificação para um coeficiente codificado de um segmento atual com base em um coeficiente previamente codificado de um segmento anterior, o coeficiente previamente codificado representando uma

15 codificado atual; e

~~um codificador de entropia (120) para codificação~~
de entropia do atual coeficiente baseado no contexto de codificação para obter um fluxo de áudio codificado,

onde o codificador de entropia (120) é adaptado
20 para codificar o coeficiente atual em unidades de uma tupla dos coeficientes espectrais e para predizer um alcance da tupla baseada no contexto de codificação;

onde o codificador de entropia (120) é adaptado
para dividir a tupla por um fator pré-determinado quando
25 necessário para encaixar um resultado da divisão em um alcance pré-determinado e para codificar um número de divisões necessárias, um lembrete de divisão e o resultado da divisão quando as tuplas não estão no alcance predito, e para codificação

de um lembrete de divisão e o resultado da divisão de outra forma;
e

onde o codificador de entropia (120) é adaptado para codificação do resultado da divisão ou as tuplas usando um índice de grupo, o índice de grupo referindo-se a um grupo de um ou mais palavras-códigos para as quais uma distribuição de probabilidade está baseado no contexto de codificação, e baseado em uma distribuição de probabilidade uniforme, um índice de elemento no caso do grupo compreender mais de uma palavra-código, o índice de elemento referindo-se a palavra-código dentro do grupo, para codificação do número de divisões pelo número de símbolos de fuga, um símbolo de fuga sendo um índice de grupo específico apenas usado para indicar a divisão e para codificação dos restantes das divisões baseadas na distribuição uniforme usando uma regra de codificação aritmética.

9. Decodificador de áudio (200) caracterizado

para decodificar um fluxo de dados codificados representando um sinal de áudio amostrado para obter uma sequência de segmentos dos coeficientes de um fluxo de áudio representando um sinal de áudio amostrado pelo uso de comprimentos diferentes de transformada tais quais segmentos de coeficientes para os quais comprimentos diferentes de transformada são usados, representa espectralmente o sinal de áudio amostrado em resoluções diferentes de frequência, compreender:

um processador (210) para obter um contexto de codificação para um coeficiente codificado de um segmento atual com base em um coeficiente previamente codificado de um segmento anterior, os segmentos anteriores e atuais correspondentes a

resoluções diferentes de frequência e comprimentos diferentes de transformada, respectivamente; e

o decodificador de entropia (220) para decodificação de entropia do atual coeficiente baseado no contexto de codificação e no fluxo de áudio codificado.

10. Codificador de áudio (200), de acordo com a reivindicação 9, **caracterizado** pelo o processador ser adaptado para obter o contexto de codificação por faixa espectral para o coeficiente atual, baseado em coeficientes espectrais vizinhos previamente decodificados no segmento anterior e opcionalmente no segmento atual.

11. Decodificador de áudio (200), de acordo com a reivindicação 10, **caracterizado** pelo o codificador de entropia (220) ser adaptado para decodificação de um índice do grupo do fluxo de áudio codificado com base em uma distribuição de probabilidade derivada a partir do contexto de codificação, onde o índice do grupo representa um grupo de uma ou mais palavras-código, e para, com base em uma distribuição de probabilidade uniforme, decodificar um índice do elemento do fluxo de áudio codificado se o índice do grupo indica um grupo composto por mais de uma palavra-código, e definir uma tupla de coeficientes espectrais do segmento atual com base no índice do grupo e o índice do elemento, obtendo assim a representação do domínio espectral nas tuplas de coeficientes espectrais.

12. Decodificador de áudio (200), de acordo com a reivindicação 11, **caracterizado** pelo codificador de entropia (220) ser adaptado para decodificação de uma sequência de símbolos do fluxo de áudio codificado com base em uma distribuição de

probabilidade derivada a partir do contexto de codificação usando um alfabeto de símbolo compreendendo um símbolo de fuga e símbolos de grupo correspondentes a um conjunto de índices disponíveis de grupo, para obter uma tupla preliminar de coeficientes espectrais baseados em um índice disponível de grupo para qual um símbolo de grupo da sequência de símbolos corresponde, e baseado no índice de elemento, e para multiplicar a tupla preliminar com o fator dependendo do número de símbolos de fuga na sequência dos símbolos para obter a tupla dos coeficientes espectrais.

10 13. Decodificador de áudio (200), de acordo com a reivindicação 12, **caracterizado** pelo codificador de entropia (220) ser adaptado para decodificação de restante de divisão do fluxo de áudio codificado baseado em uma distribuição de probabilidade uniforme usando uma regra de codificação aritmética e para adição
15 do restante a tupla preliminar multiplicada para obter a tupla dos coeficientes espectrais.

14. Decodificador de áudio (200), de acordo com uma das reivindicações anteriores, **caracterizado** pelo processador (210) e o codificador de entropia (220) serem configurados para
20 operar baseados em um downsampling de coeficientes espectrais dos segmentos anteriores, quando o segmento anterior exibe uma melhor resolução espectral do que o segmento atual e/ou onde o processador (210) e o codificador de entropia (220) estão configurados para operar baseados em um upsampling de coeficientes
25 espectrais do segmento anterior, quando o segmento anterior exibe uma resolução espectral mais rígida que o segmento atual.

15. Decodificador de áudio (200), **caracterizado** por ser para decodificar um fluxo de áudio codificado para obter

segmentos de coeficientes representando diferentes resoluções de tempo ou frequência de um sinal de áudio amostrado, que compreende:

um processador (210) para obter um contexto de
5 codificação para um coeficiente decodificado de um segmento atual com base em um coeficiente previamente decodificado de um segmento anterior, o coeficiente anteriormente decodificado representando uma resolução diferente de frequência e tempo do que o coeficiente atualmente decodificado; e

10 o decodificador de entropia (220) para decodificação de entropia do atual coeficiente baseado no contexto de codificação e no fluxo de áudio codificado.

onde o processador (210) é adaptado para obter o contexto de codificação baseado no coeficiente anterior, os
15 coeficientes formando uma representação espectral de um sinal de áudio representado por um fluxo de áudio em uma resolução espectral variando entre os segmentos;

onde o processador é adaptado para obter o contexto de codificação por faixa espectral para o coeficiente
20 atual, baseado em coeficientes espectrais vizinhos previamente decodificados no segmento anterior e opcionalmente no segmento atual; e

onde o codificador de entropia (220) é adaptado para decodificação de um índice do grupo do fluxo de áudio
25 codificado com base em uma distribuição de probabilidade obtido a partir do contexto de codificação, onde o índice do grupo representa um grupo de uma ou mais palavras-código, e para, com base em uma distribuição de probabilidade uniforme, decodificar um

índice do elemento do fluxo de áudio codificado se o índice do grupo indica um grupo composto por mais de uma palavra-código, e definir uma tupla de coeficientes espectrais do segmento atual com base no índice do grupo e o índice do elemento, obtendo assim a
 5 representação do domínio espectral nas tuplas de coeficientes espectrais.

16. Método para decodificar um fluxo de dados codificados representando um sinal de áudio amostrado, **caracterizado** por ser obter uma sequência de segmentos dos
 10 coeficientes de um fluxo de áudio representando um sinal de áudio amostrado pelo uso de comprimentos diferentes de transformada tais quais segmentos de coeficientes para os quais comprimentos diferentes de transformada são usados, representa espectralmente o sinal de áudio amostrado em resoluções diferentes de frequência,
 15 - compreender: - - - - -

~~para obter um contexto de codificação para um~~
 coeficiente codificado atualmente de um segmento atual com base em um coeficiente previamente codificado de um segmento anterior, os segmentos anteriores e atuais correspondentes a resoluções
 20 diferentes de frequência e comprimentos diferentes de transformada, respectivamente; e

decodificação de entropia do atual coeficiente baseado no contexto de codificação e no fluxo de áudio codificado.

17. Programa de computador, **caracterizado** por
 25 compreender um código de programa para realizar um dos métodos das reivindicações 7 ou 15, quando for executado em um computador ou processador.

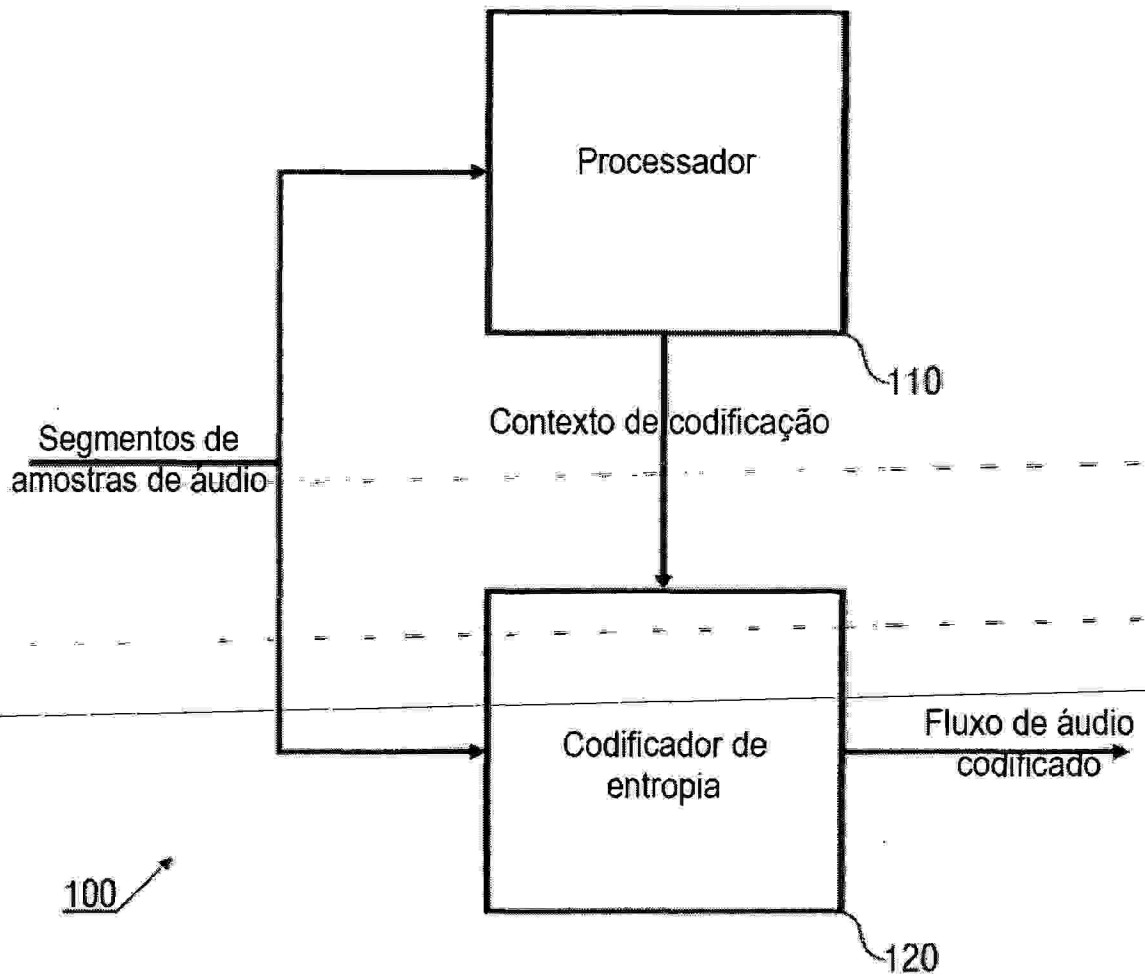


FIG 1

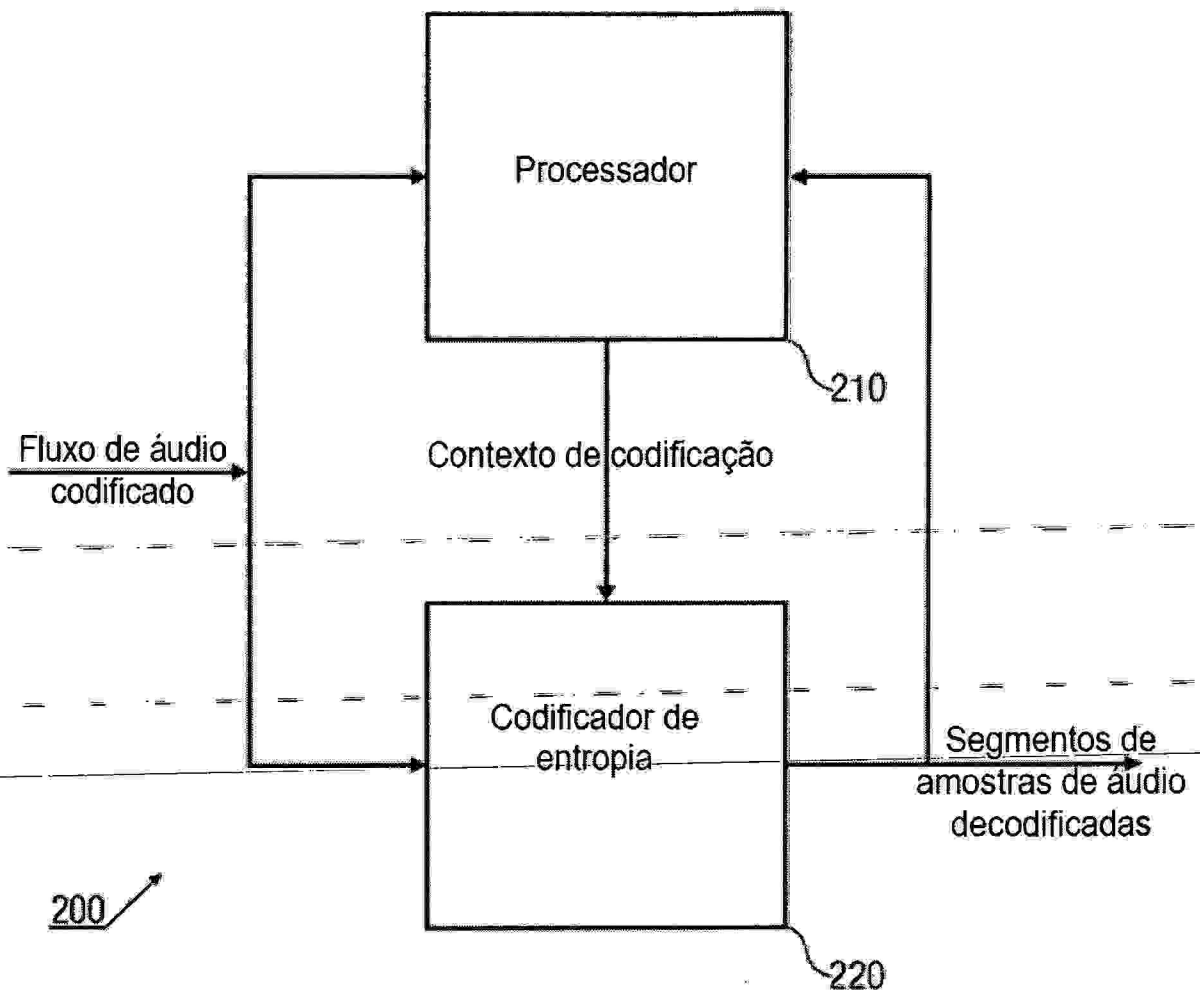


FIG 2

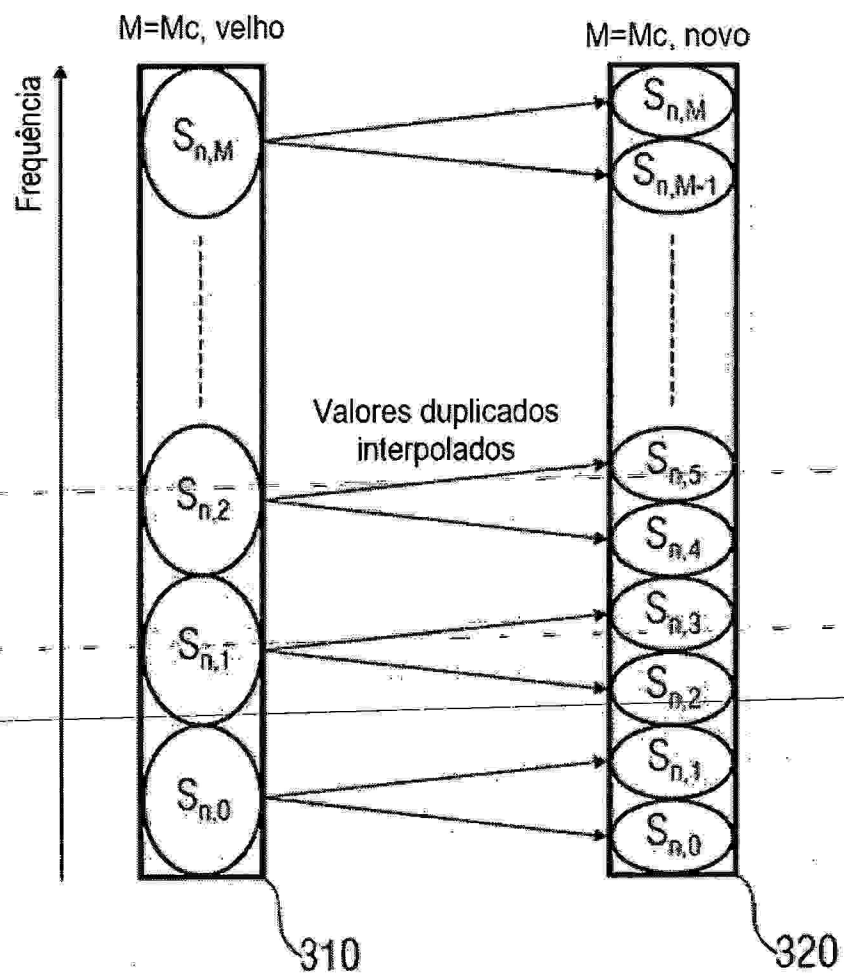


FIG 3

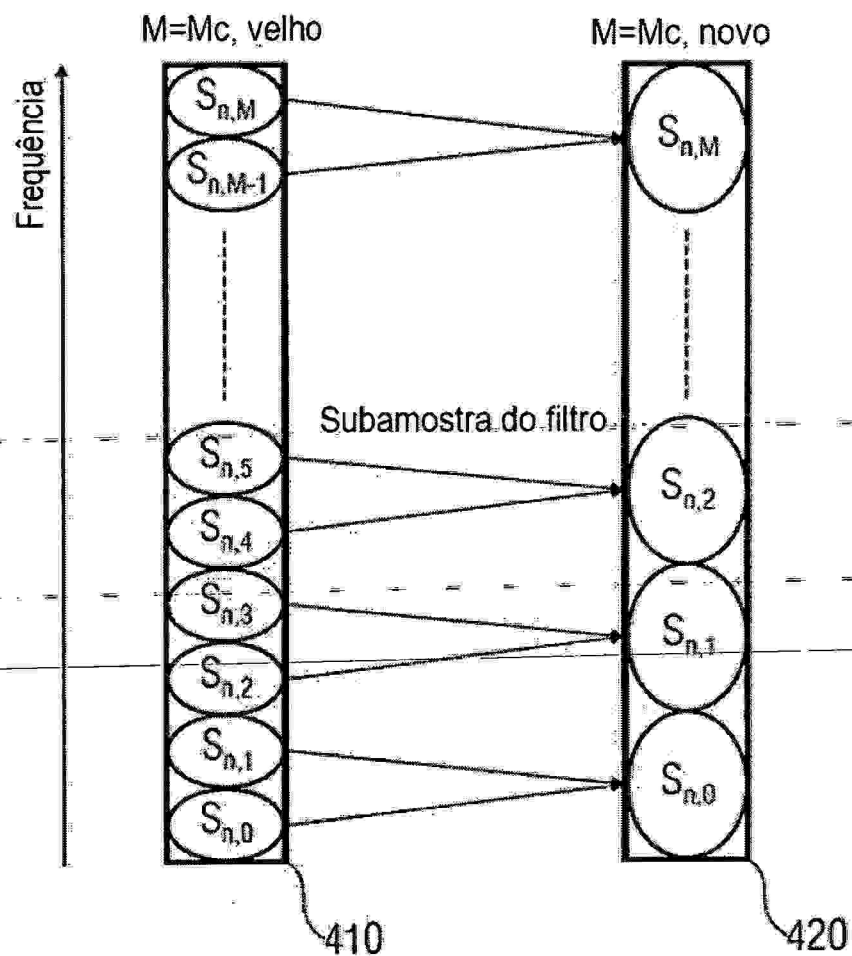


FIG 4

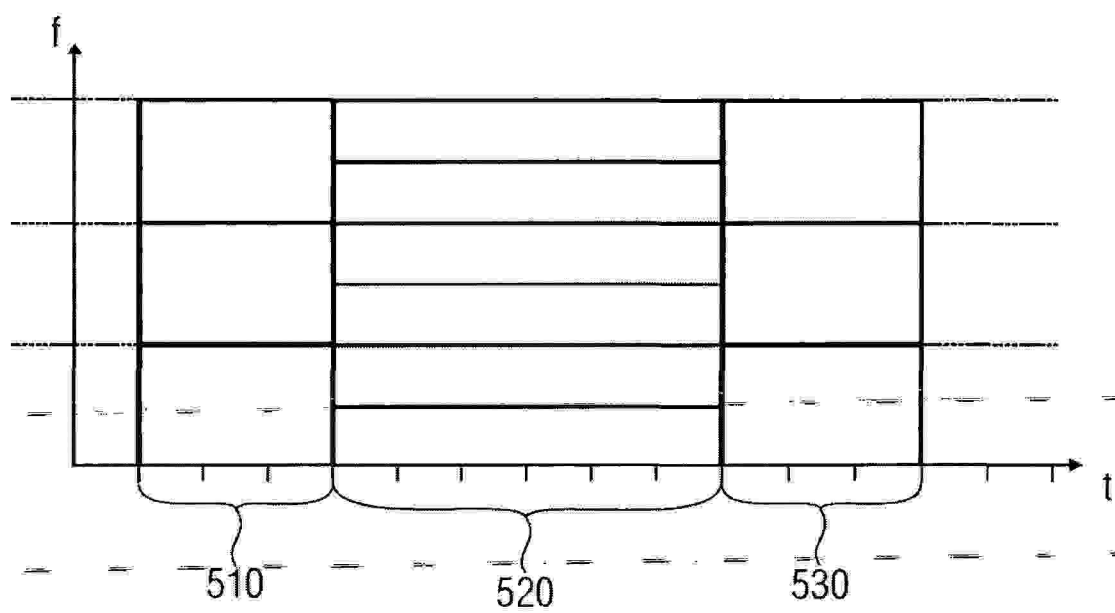


FIG 5

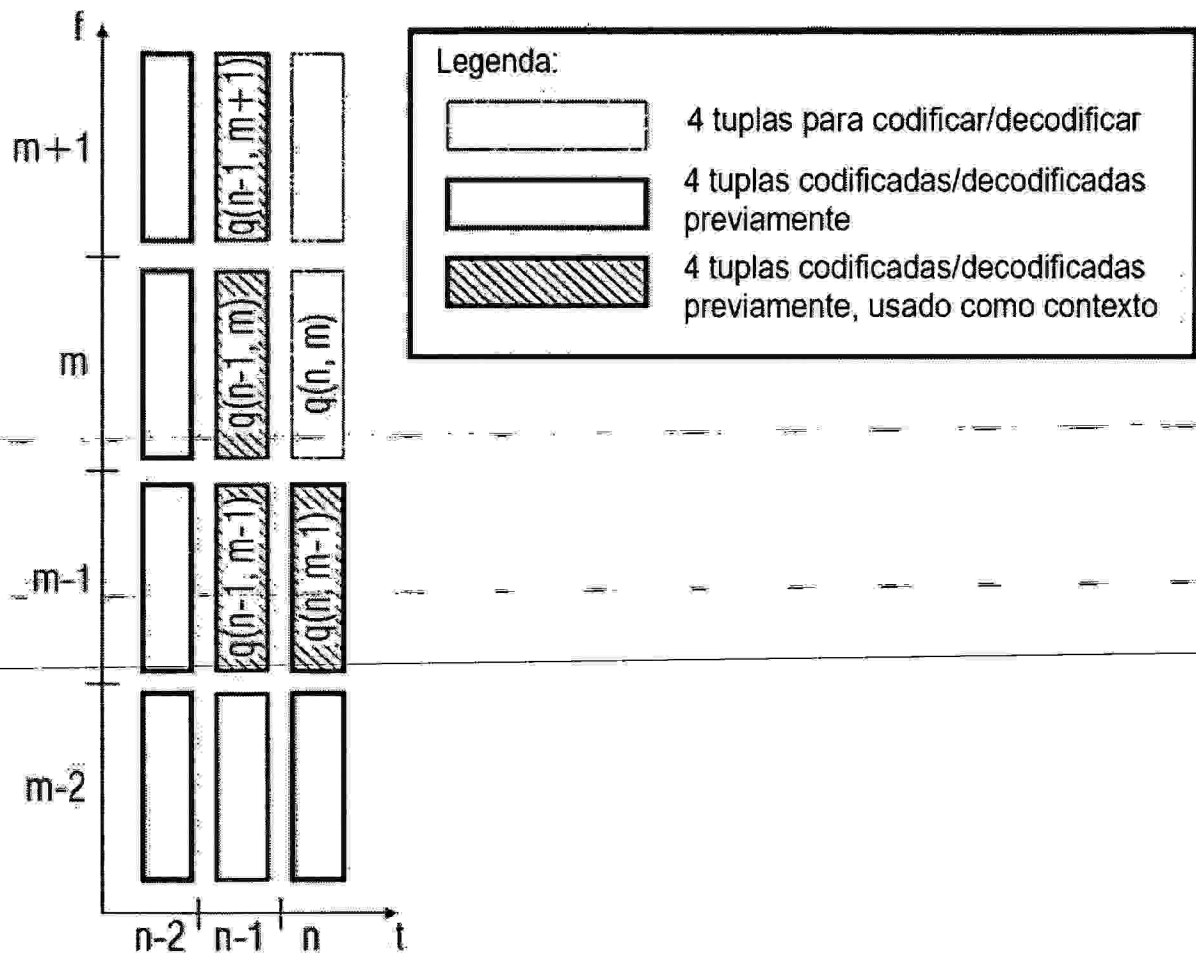


FIG 6

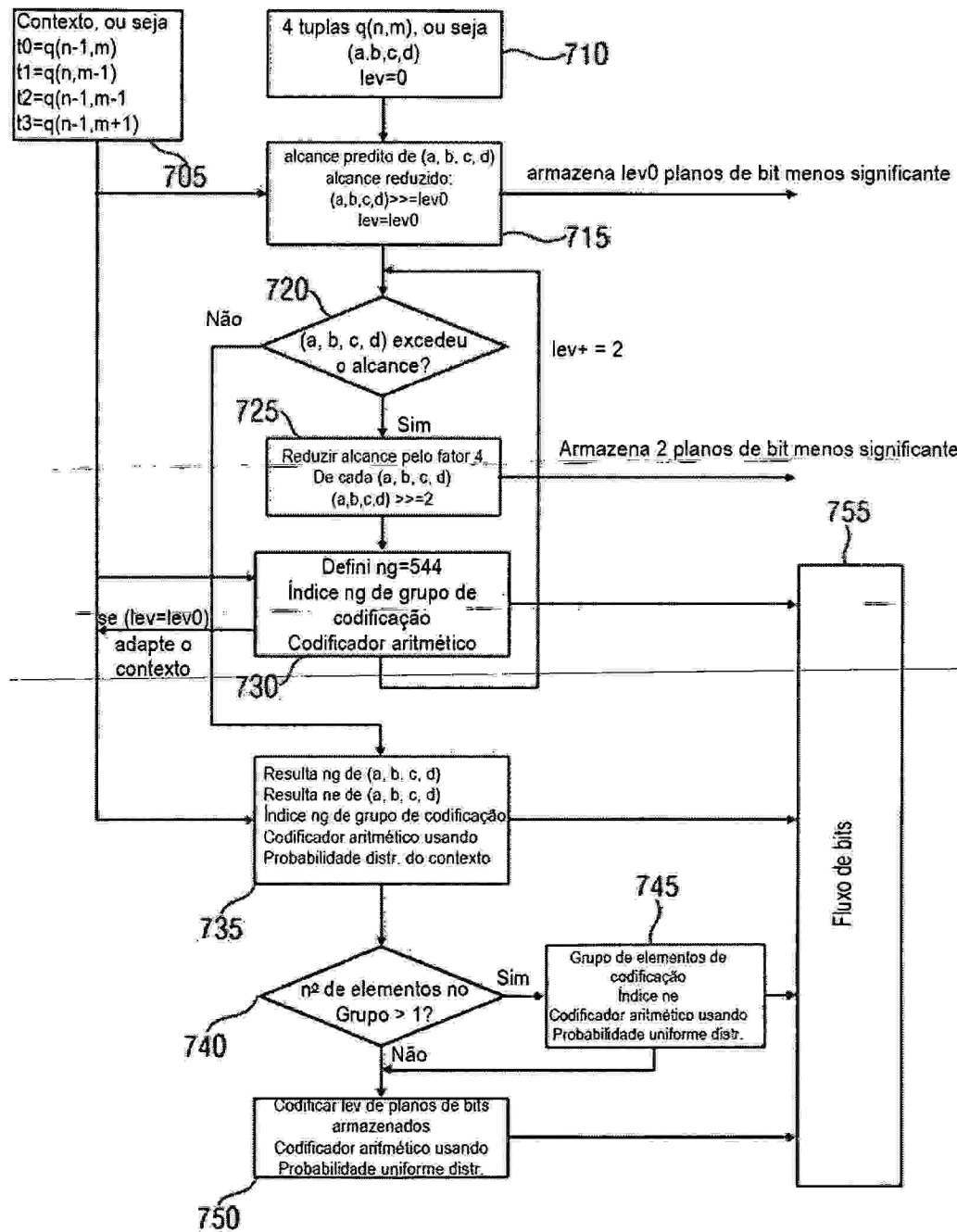


FIG 7A

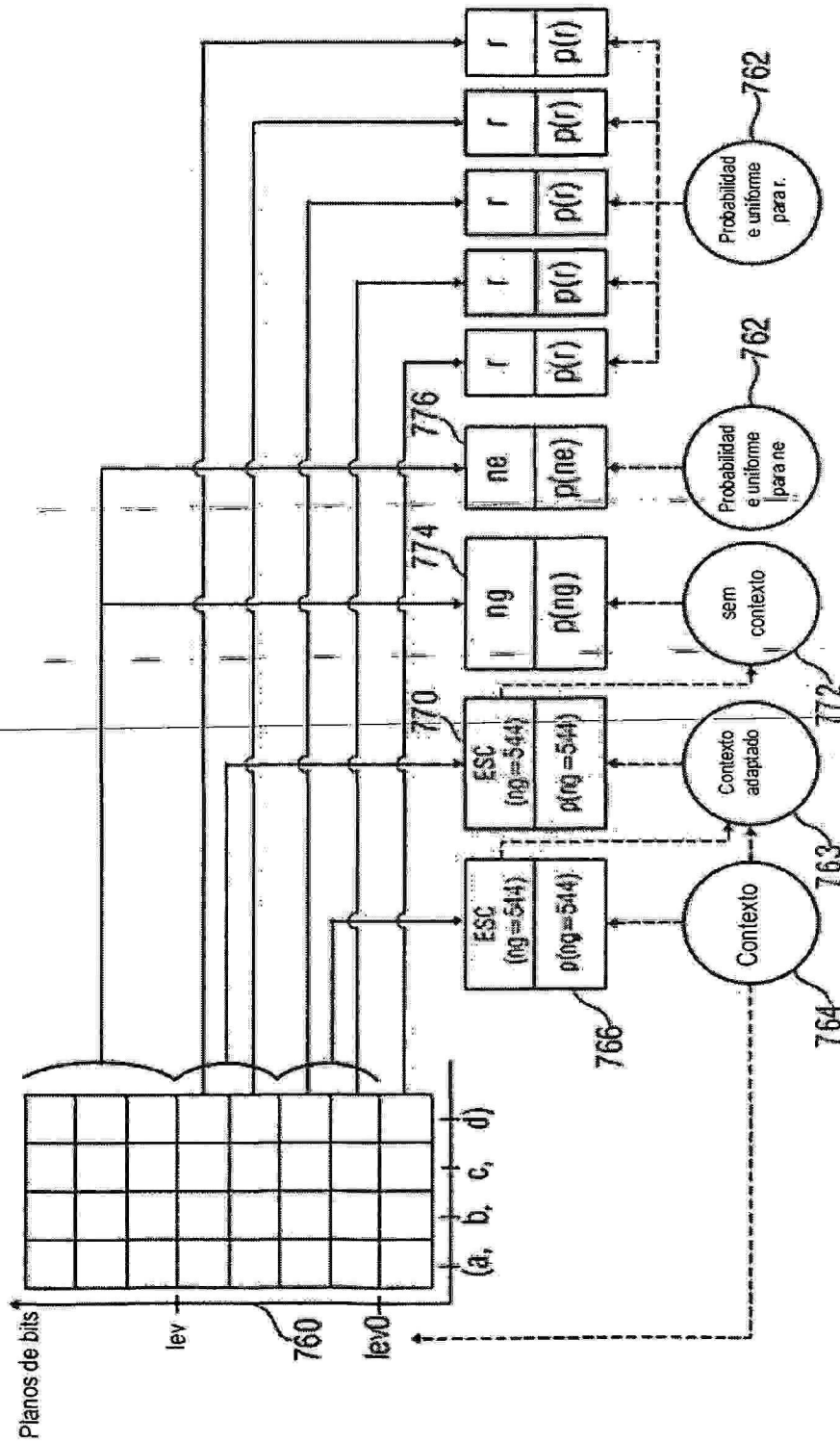


FIG 7B

9/11

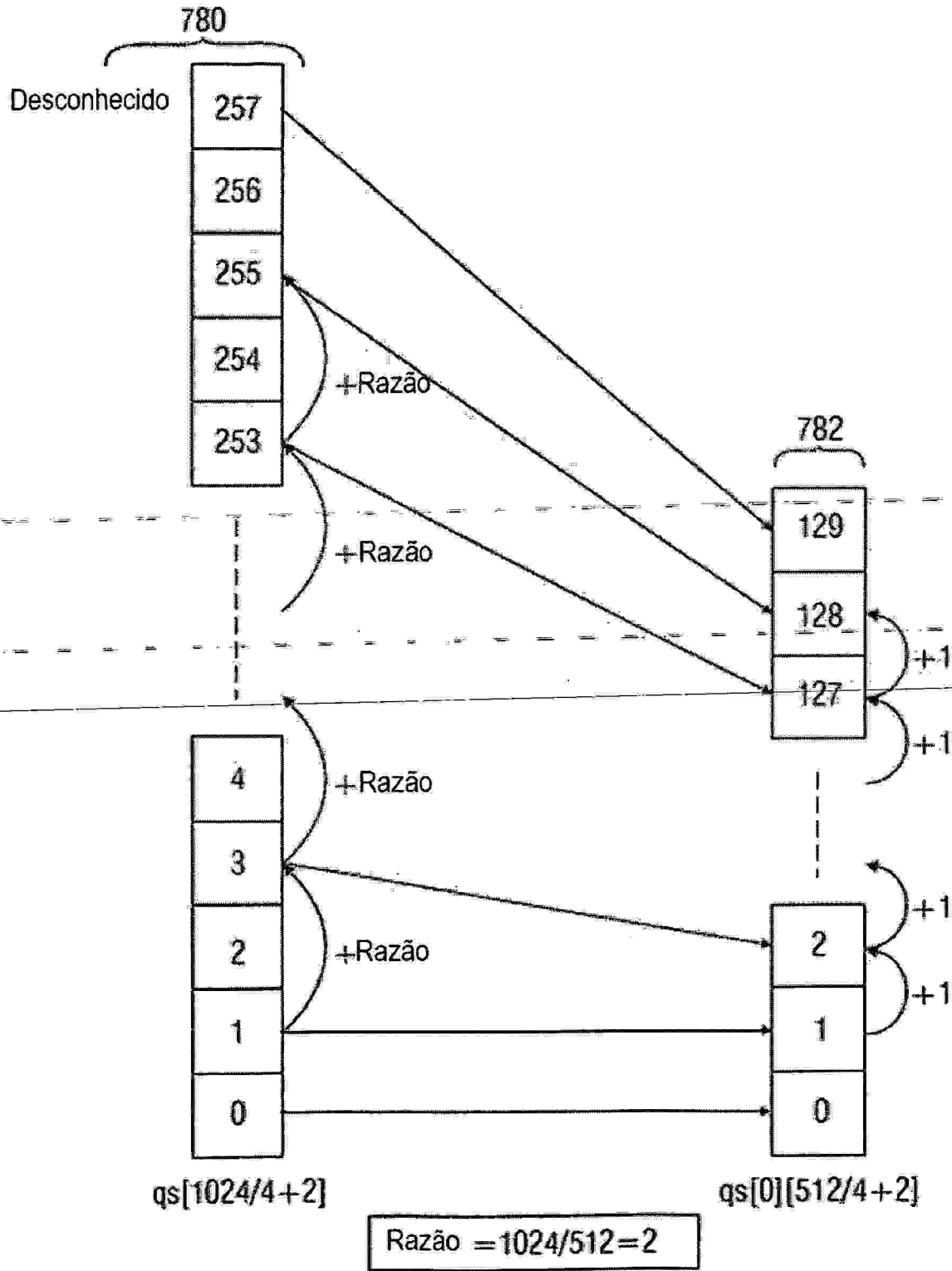


FIG 7C

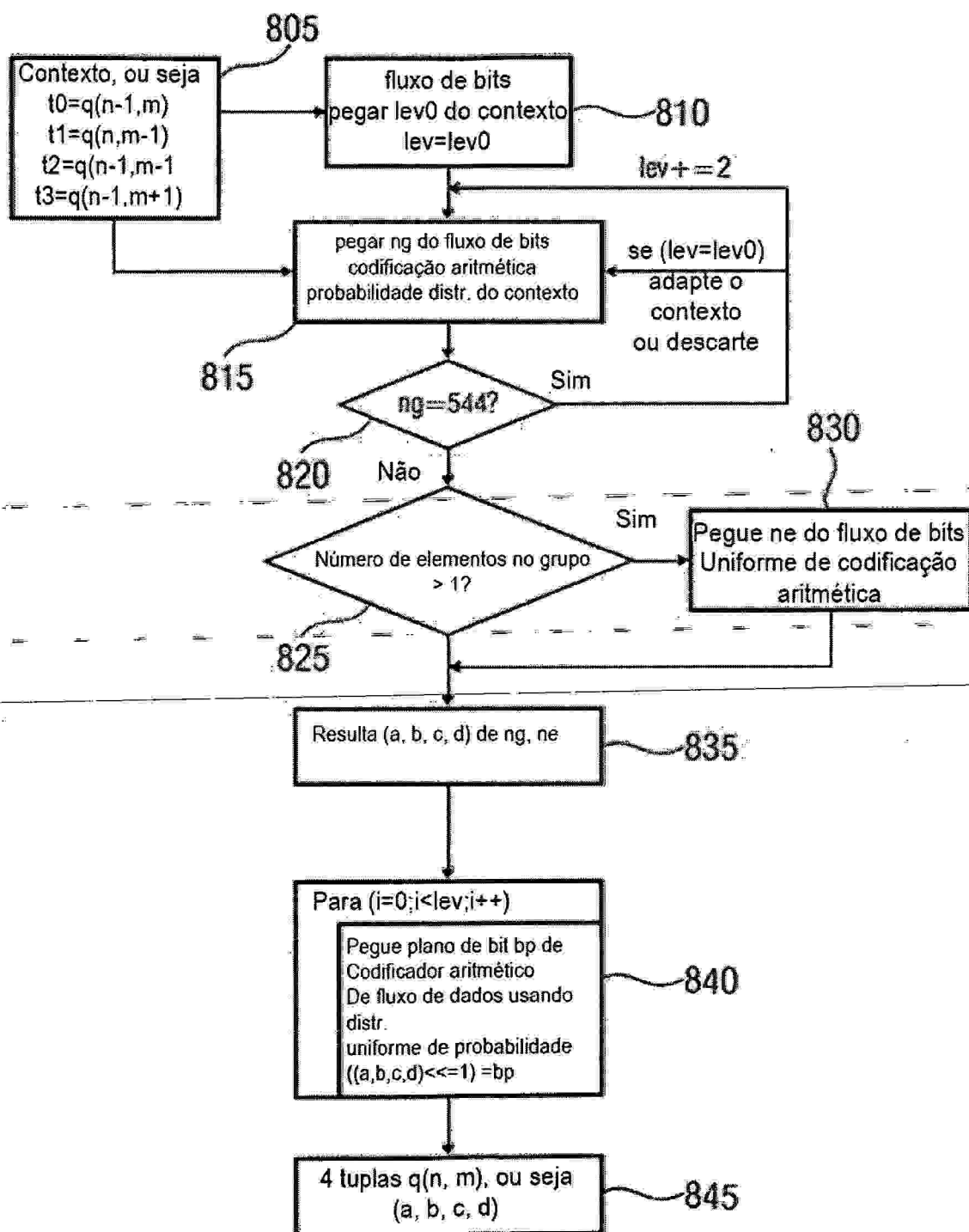


FIG 8

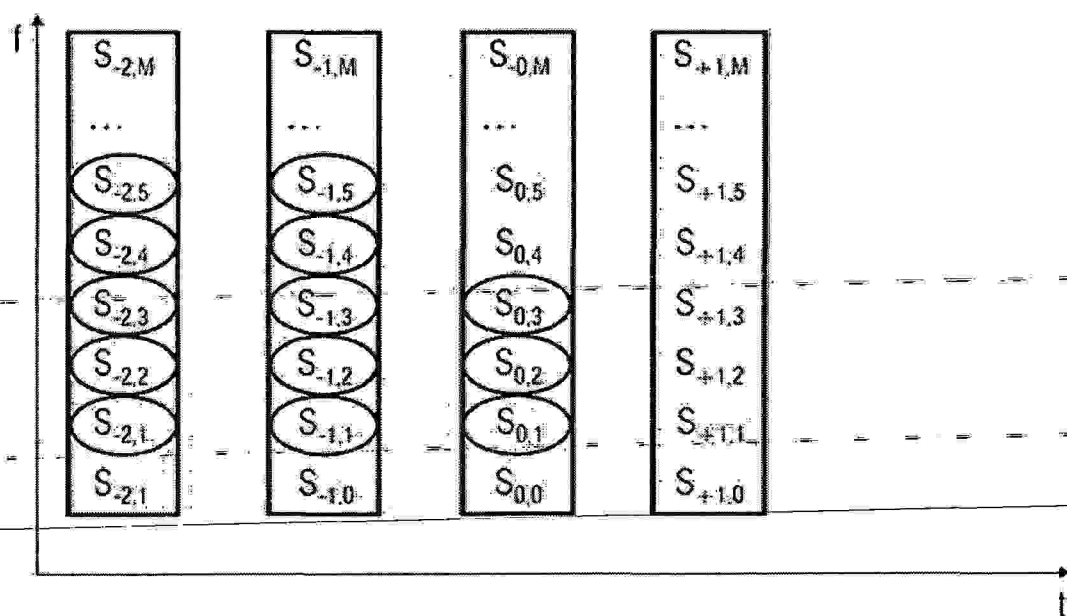


FIG 9
(ESTADO DA TÉCNICA)

RESUMO

"CODIFICADOR DE ÁUDIO E DECODIFICADOR DE ÁUDIO"

Um decodificador de áudio (110) para segmentos de codificação de coeficientes, os segmentos de coeficientes representando resoluções diferentes de tempo e frequência de um sinal de áudio amostrado, o codificador de áudio (100) compreendendo um processador (110) para derivar um contexto de codificação para o coeficiente codificado atual baseado em um coeficiente codificado anterior de um segmento anterior, o coeficiente previamente codificado representando uma diferente resolução de tempo ou frequência do que o coeficiente codificado atual. O codificador de áudio (100) compreende ainda um codificador de entropia (120) para codificação de entropia do atual coeficiente baseado no contexto de codificação para obter um

15 fluxo de áudio codificado.
